The Fault-Tolerant Cluster-Sending Problem

Jelle Hellings[1] and Mohammad Sadoghi[2]

[1] McMaster University, 1280 Main St. W., Hamilton, ON L8S 4L7, Canada
[2] Exploratory Systems Lab, Department of Computer Science, University of California, Davis, USA

Abstract The emergence of blockchains is fueling the development of resilient data management systems that can deal with *Byzantine failures* due to crashes, bugs, or even malicious behavior. As traditional resilient systems lack the scalability required for modern data, several recent systems explored using *sharding*. Enabling these sharded designs requires two basic primitives: a primitive to reliably make decisions within a cluster and a primitive to reliably communicate between clusters. Unfortunately, such communication has not yet been formally studied.
In this work, we improve on this situation by formalizing the *cluster-sending problem*: the problem of sending a message from one resilient system to another in a fault-tolerant manner. We also establish lower bounds on the complexity of cluster-sending under both crashes and Byzantine failures. Finally, we present *worst-case optimal* cluster-sending protocols that meet these lower bounds in practical settings. As such, our work provides a strong foundation for the future development of sharded resilient data management systems.

Keywords: Byzantine Failures · Sharding · Message Sending · Communication Lower Bounds · Worst-Case Optimal Communication

1 Introduction

The emergence of blockchain technology is fueling interest in the development of new data management systems that can manage data between fully-independent parties (*federated data management)* and can provide services continuously, even during *Byzantine failures* (e.g., network failure, hardware failure, software failure, or malicious attacks) [5,13,14,18,20,22]. Recently, this has led to the development of several resilient data management systems based on *permissioned blockchain technology* [6,7,8,9].

Unfortunately, systems based on traditional fully-replicated *consensus-based* permissioned blockchain technology lack the scalability required for modern data management. Consequently, several recent systems have proposed to combine sharding with consensus-based designs (e.g., `AHL` [3], `ByShard` [10], and `Chainspace` [1]). These systems all follow a familiar sharded design: the data is split up into individual pieces called *shards* and each shard is managed by different independent blockchain-driven clusters. To illustrate the benefits of sharding, consider a system with a *sharded design* in which data is kept in *local Byzantine*

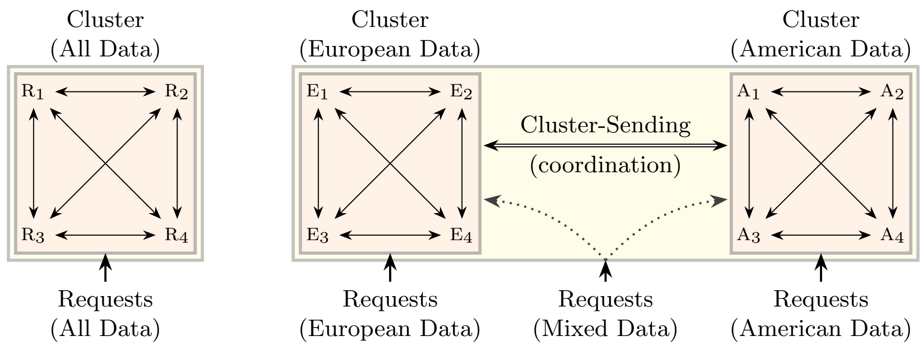

Figure 1. *Left*, a traditional fully-replicated resilient system in which all four replicas each hold all data. *Right*, a *sharded* design in which each resilient cluster of four replicas holds only a part of the data.

fault-tolerant clusters, e.g., as sketched in Figure 1 by storing data relevant to American customers on systems located in the United States, whereas systems located in Europe contain data relevant to European customers. Compared to the traditional fully-replicated design of blockchain systems, this sharded design will improve *storage scalability* by distributing data storage and improve *performance scalability* by enabling concurrent transaction processing, e.g., transactions on American and European data can be performed independently of each other.

At the core of any sharded data processing system are two crucial primitives [17]. First, individual shards need primitives to independently make *decisions*, e.g., to execute transactions that only affect data held within that shard. In the setting where each shard is a fault-tolerant cluster, such per-shard decision making is formalized by the well-known *consensus problem*, which can be solved by practical consensus protocols such as `Pbft` [2]. Second, shards need primitives to *communicate* between each other, e.g., to coordinate the execution of transactions that affect data held by multiple shards. Unfortunately, even though inter-shard communication is a fundamental basic primitive, it has not yet been studied in much detail. Indeed, existing sharded blockchain-inspired data processing systems typically use expensive ad-hoc techniques to enable coordination between shards (e.g., `Chainspace` [1] uses expensive all-to-all broadcasts).

In this work, we improve on this situation by formalizing the problem of inter-shard communication in permissioned fault-tolerant systems: the *cluster-sending problem*. In specific, we fully formalize the cluster-sending problem in Section 2. Then, in Section 3, we prove strict *lower bounds on the complexity* of the cluster-sending problem that are *linear* in terms of the number of messages (when faulty replicas only crash) and in terms of the number of signatures (when faulty replicas can be malicious and messages are signed via public-key cryptography). Next, in Sections 4 and 5, we introduce *bijective sending* and *partitioned bijective sending*, powerful techniques to provide *worst-case optimal* cluster-sending between clusters of roughly the same size (bijective sending) and

Protocol	System	Robustness	Messages	(*size*)
BS-cs	Omit	$\mathbf{n}_{\mathcal{C}_1}, \mathbf{n}_{\mathcal{C}_2} > \mathbf{f}_{\mathcal{C}_1} + \mathbf{f}_{\mathcal{C}_2}$	$\mathbf{f}_{\mathcal{C}_1} + \mathbf{f}_{\mathcal{C}_2} + 1$ (optimal)	$\mathcal{O}(\lVert v \rVert)$
BS-rs	Byzantine, RS	$\mathbf{n}_{\mathcal{C}_1}, \mathbf{n}_{\mathcal{C}_2} > 2\mathbf{f}_{\mathcal{C}_1} + \mathbf{f}_{\mathcal{C}_2}$	$2\mathbf{f}_{\mathcal{C}_1} + \mathbf{f}_{\mathcal{C}_2} + 1$ (optimal)	$\mathcal{O}(\lVert v \rVert)$
BS-cs	Byzantine, CS	$\mathbf{n}_{\mathcal{C}_1}, \mathbf{n}_{\mathcal{C}_2} > \mathbf{f}_{\mathcal{C}_1} + \mathbf{f}_{\mathcal{C}_2}$	$\mathbf{f}_{\mathcal{C}_1} + \mathbf{f}_{\mathcal{C}_2} + 1$ (optimal)	$\mathcal{O}(\lVert v \rVert)$
PBS-cs	Omit	$\mathbf{n}_{\mathcal{C}_1} > 3\mathbf{f}_{\mathcal{C}_1}$, $\mathbf{n}_{\mathcal{C}_2} > 3\mathbf{f}_{\mathcal{C}_2}$	$\mathcal{O}(\max(\mathbf{n}_{\mathcal{C}_1}, \mathbf{n}_{\mathcal{C}_2}))$ (optimal)	$\mathcal{O}(\lVert v \rVert)$
PBS-rs	Byzantine, RS	$\mathbf{n}_{\mathcal{C}_1} > 4\mathbf{f}_{\mathcal{C}_1}$, $\mathbf{n}_{\mathcal{C}_2} > 4\mathbf{f}_{\mathcal{C}_2}$	$\mathcal{O}(\max(\mathbf{n}_{\mathcal{C}_1}, \mathbf{n}_{\mathcal{C}_2}))$ (optimal)	$\mathcal{O}(\lVert v \rVert)$
PBS-cs	Byzantine, CS	$\mathbf{n}_{\mathcal{C}_1} > 3\mathbf{f}_{\mathcal{C}_1}$, $\mathbf{n}_{\mathcal{C}_2} > 3\mathbf{f}_{\mathcal{C}_2}$	$\mathcal{O}(\max(\mathbf{n}_{\mathcal{C}_1}, \mathbf{n}_{\mathcal{C}_2}))$ (optimal)	$\mathcal{O}(\lVert v \rVert)$
Chainspace [1]	Byzantine, CS	$\mathbf{n}_{\mathcal{C}_1} > 3\mathbf{f}_{\mathcal{C}_1}$, $\mathbf{n}_{\mathcal{C}_2} > 3\mathbf{f}_{\mathcal{C}_2}$	$\mathcal{O}(\mathbf{n}_{\mathcal{C}_1} \cdot \mathbf{n}_{\mathcal{C}_2})$	$\mathcal{O}(\lVert v \rVert)$

Figure 2. Overview of cluster-sending protocols that sends a value v of size $\lVert v \rVert$ from cluster $\mathcal{C}_1$ to cluster $\mathcal{C}_2$. Cluster $\mathcal{C}_i$, $i \in \{1, 2\}$, has $\mathbf{n}_{\mathcal{C}_i}$ replicas of which $\mathbf{f}_{\mathcal{C}_i}$ are faulty. The protocol names (first column) indicate the main principle the protocol relies on (BS for *bijective sending*, and PBS for *partitioned bijective sending*), and the specific variant the protocol is designed for (variant -cs is designed to use cluster signing, and variant -rs is designed to use replica signing). The system column describe the type of Byzantine behavior the protocol must deal with ("Omit" for systems in which Byzantine replicas can drop messages, and "Byzantine" for systems in which Byzantine replicas have arbitrary behavior) and the signature scheme present in the system ("RS" is shorthand for *replica signing*, and "CS" is shorthand for *cluster signing*).

of arbitrary sizes (partitioned bijective sending). Finally, in Section 6, we evaluate the behavior of the proposed cluster-sending protocols via an in-depth evaluation. In this evaluation, we show that our worst-case optimal cluster-sending protocols have exceptionally low communication costs in comparison with existing ad-hoc approaches from the literature. A full overview of all environmental conditions in which we study the cluster-sending problem and the corresponding worst-case optimal cluster-sending protocols we propose can be found in Figure 2.

Our cluster-sending problem is closely related to cross-chain coordination in *permissionless blockchains* such as Bitcoin [16] and Ethereum [21], e.g., as provided via atomic swaps [11], atomic commitment [24], and cross-chain deals [12]. Unfortunately, such permissionless solutions are not fit for a permissioned environment. Although *cluster-sending* can be solved using well-known permissioned techniques such as consensus, interactive consistency, Byzantine broadcasts, and message broadcasting [2,4], the best-case costs for these primitives are much higher than the worst-case costs of our cluster-sending protocols, making them unsuitable for cluster-sending. As such, the cluster-sending problem is an independent problem and our initial results on this problem provide novel directions for the design and implementation of high-performance resilient data management systems.

2 Formalizing the Cluster-Sending Problem

A *cluster* $\mathcal{C}$ is a set of replicas. We write $\mathsf{f}(\mathcal{C}) \subseteq \mathcal{C}$ to denote the set of *faulty replicas* in $\mathcal{C}$ and $\mathsf{nf}(\mathcal{C}) = \mathcal{C} \setminus \mathsf{f}(\mathcal{C})$ to denote the set of *non-faulty replicas* in $\mathcal{C}$. We write $\mathbf{n}_{\mathcal{C}} = |\mathcal{C}|$, $\mathbf{f}_{\mathcal{C}} = |\mathsf{f}(\mathcal{C})|$, and $\mathbf{nf}_{\mathcal{C}} = |\mathsf{nf}(\mathcal{C})|$ to denote the number of replicas, faulty replicas, and non-faulty replicas in the cluster, respectively. We

	Ping round-trip times (ms)						*Bandwidth* (Mbit/s)					
	O	I	M	B	T	S	O	I	M	B	T	S
Oregon (O)	≤ 1	38	65	136	118	161	7998	669	371	194	188	136
Iowa (I)		≤ 1	33	98	153	172		10004	752	243	144	120
Montreal (M)			≤ 1	82	186	202			7977	283	111	102
Belgium (B)				≤ 1	252	270				9728	79	66
Taiwan (T)					≤ 1	137					7998	160
Sydney (S)						≤ 1						7977

Figure 3. Real-world communication costs in Google Cloud, using clusters of `n1` machines deployed in six different regions, in terms of the ping round-trip times (which determines *latency*) and bandwidth (which determines *throughput*). These measurements are reproduced from Gupta et al. [8].

extend the notations $\mathsf{f}(\cdot)$, $\mathsf{nf}(\cdot)$, $\mathbf{n}_{(\cdot)}$, $\mathbf{f}_{(\cdot)}$, and $\mathbf{nf}_{(\cdot)}$ to arbitrary sets of replicas. We assume that all replicas in each cluster have a predetermined order (e.g., on identifier or on public address), which allows us to deterministically select any number of replicas in a unique order from each cluster. In this work, we consider faulty replicas that can *crash*, *omit* messages, or behave *Byzantine*. A *crashing replica* executes steps correctly up till some point, after which it does not execute anything. An *omitting replica* executes steps correctly, but can decide to not send a message when it should or decide to ignore messages it receives. A *Byzantine replica* can behave in arbitrary, possibly coordinated and malicious, manners.

A *cluster system* $\mathfrak{S}$ is a finite set of clusters such that communication between replicas in a cluster is *local* and communication between clusters is *non-local*. We assume that there is no practical bound on local communication (e.g., within a single data center), while global communication is limited, costly, and to be avoided (e.g., between data centers in different continents). If $\mathcal{C}_1, \mathcal{C}_2 \in \mathfrak{S}$ are distinct clusters, then we assume that $\mathcal{C}_1 \cap \mathcal{C}_2 = \emptyset$: no replica is part of two distinct clusters. Our abstract model of a cluster system—in which we distinguish between unbounded local communication and costly global communication—is supported by practice. E.g., the ping round-trip time and bandwidth measurements of Figure 3 imply that message latencies between clusters are at least 33–270 times higher than within clusters, while the maximum throughput is 10–151 times lower, both implying that communication between clusters is *up-to-two orders of magnitude* more costly than communication within clusters.

Definition 1. *Let $\mathfrak{S}$ be a system and $\mathcal{C}_1, \mathcal{C}_2 \in \mathfrak{S}$ be two clusters with non-faulty replicas ($\mathsf{nf}(\mathcal{C}_1) \neq \emptyset$ and $\mathsf{nf}(\mathcal{C}_2) \neq \emptyset$). The* cluster-sending problem *is the problem of sending a value v from $\mathcal{C}_1$ to $\mathcal{C}_2$ such that: **(1.)** all non-faulty replicas in $\mathcal{C}_2$* receive *the value v; **(2.)** all non-faulty replicas in $\mathcal{C}_1$* confirm *that the value v was received by all non-faulty replicas in $\mathcal{C}_2$; and **(3.)** non-faulty replicas in $\mathcal{C}_2$ can only receive a value v if all non-faulty replicas in $\mathcal{C}_1$* agree *upon sending v.*

In the following, we assume *asynchronous reliable communication*: all messages sent by non-faulty replicas eventually arrive at their destination. None of the protocols we propose rely on message delivery timings for their correctness. We assume that communication is *authenticated*: on receipt of a message m from replica $\textsc{r} \in \mathcal{C}$, one can determine that $\textsc{r}$ did send m if $\textsc{r} \in \mathsf{nf}(\mathcal{C})$ and if $\textsc{r} \in \mathsf{nf}(\mathcal{C})$, then one can only determine that m was sent by $\textsc{r}$ if $\textsc{r}$ did send m. Hence, faulty replicas are only able to impersonate each other. We study the *cluster-sending problem* for Byzantine systems in two types of environments:

1. A system provides *replica signing* if every replica $\textsc{r}$ can *sign* arbitrary messages m, resulting in a certificate $\langle m \rangle_{\textsc{r}}$. These certificates are non-forgeable and can be constructed only if $\textsc{r}$ cooperates in constructing them. Based on only the certificate $\langle m \rangle_{\textsc{r}}$, anyone can verify that m was supported by $\textsc{r}$.
2. A system provides *cluster signing* if it is equipped with a *signature scheme* that can be used to *cluster-sign* arbitrary messages m, resulting in a certificate $\langle m \rangle_{\mathcal{C}}$. These certificates are non-forgeable and can be constructed whenever all non-faulty replicas in $\mathsf{nf}(\mathcal{C})$ cooperate in constructing them. Based on only the certificate $\langle m \rangle_{\mathcal{C}}$, anyone can verify that m was originally supported by all non-faulty replicas in $\mathcal{C}$.

In practice, *replica signing* can be implemented using *digital signatures*, which rely on a public-key cryptography infrastructure [15], and *cluster signing* can be implemented using threshold signatures, which are available for some public-key cryptography infrastructures [19]. Let m be a message, $\mathcal{C} \in \mathfrak{S}$ a cluster, and $\textsc{r} \in \mathcal{C}$ a replica. We write $\|v\|$ to denote the size of any arbitrary value v. We assume that the size of certificates $\langle m \rangle_{\textsc{r}}$, obtained via replica signing, and certificates $\langle m \rangle_{\mathcal{C}}$, obtained via cluster signing, are both linearly upper-bounded by $\|m\|$. More specifically, $\|(m, \langle m \rangle_{\textsc{r}})\| = \mathcal{O}(\|m\|)$ and $\|(m, \langle m \rangle_{\mathcal{C}})\| = \mathcal{O}(\|m\|)$.

When necessary, we assume that replicas in each cluster $\mathcal{C} \in \mathfrak{S}$ can reach agreement on a value using an off-the-shelf *consensus protocol* [2,23]. In general, these protocols require $\mathbf{n}_{\mathcal{C}} > 2\mathbf{f}_{\mathcal{C}}$ (crash failures) or $\mathbf{n}_{\mathcal{C}} > 3\mathbf{f}_{\mathcal{C}}$ (Byzantine failures), which we assume to be the case for all *sending clusters*. Finally, in this paper we use the notation $i \operatorname{sgn} j$, with $i, j \geq 0$ and sgn the sign function, to denote i if $j > 0$ and 0 otherwise.

3 Lower Bounds for Cluster-Sending

In the previous section, we formalized the cluster-sending problem. The cluster-sending problem can be solved intuitively using *message broadcasts* (e.g., as used by `Chainspace` [1]), a principle technique used in the implementation of Byzantine primitives such as consensus and interactive consistency to assure that all non-faulty replicas reach the same conclusions. Unfortunately, broadcast-based protocols have a high communication cost that is quadratic in the size of the clusters involved. To determine whether we can do better than broadcasting, we will study the *lower bound* on the communication cost for any protocol solving the cluster-sending problem.

First, we consider systems with only crash failures, in which case we can lower bound the number of messages exchanged. As systems with omit failures or Byzantine failures can behave as-if they have only crash failures, these lower bounds apply to all environments. Any lower bound on the number of messages

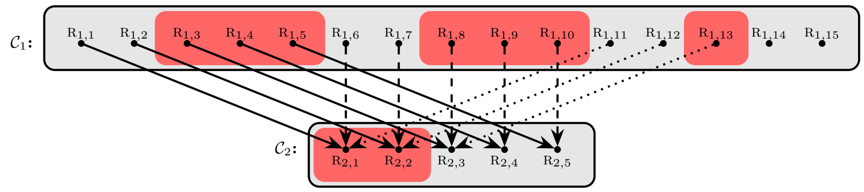

Figure 4. A run of a protocol that sends messages from $\mathcal{C}_1$ and $\mathcal{C}_2$. The protocol P sends 13 messages, which is one message short of guaranteeing successful cluster-sending. Hence, to thwart cluster-sending in this particular run we can crash (highlighted using a red background) $\mathbf{f}_{\mathcal{C}_1} = 7$ and $\mathbf{f}_{\mathcal{C}_2} = 2$ replicas in $\mathcal{C}_1$ and $\mathcal{C}_2$, respectively.

exchanged is determined by the maximum number of messages that can get *lost* due to crashed replicas that do not send or receive messages. If some replicas need to send or receive *multiple* messages, the capabilities of crashed replicas to lose messages is likewise multiplied, as the following example illustrates.

Example 1. Consider a system $\mathfrak{S}$ with clusters $\mathcal{C}_1, \mathcal{C}_2 \in \mathfrak{S}$ such that $\mathbf{n}_{\mathcal{C}_1} = 15$, $\mathbf{f}_{\mathcal{C}_1} = 7$, $\mathbf{n}_{\mathcal{C}_2} = 5$, and $\mathbf{f}_{\mathcal{C}_2} = 2$. We assume that $\mathfrak{S}$ only has crash failures and that the cluster $\mathcal{C}_1$ wants to send value v to $\mathcal{C}_2$. We will argue that any correct cluster-sending protocol P needs to send at least 14 messages in the worst case, as we can always assure that up-to-13 messages will get lost by crashing $\mathbf{f}_{\mathcal{C}_1}$ replicas in $\mathcal{C}_1$ and $\mathbf{f}_{\mathcal{C}_2}$ replicas in $\mathcal{C}_2$.

Consider the messages of a protocol P that wants to send only 13 messages from $\mathcal{C}_1$ to $\mathcal{C}_2$, e.g., the run in Figure 4. Notice that $13 > \mathbf{n}_{\mathcal{C}_2}$. Hence, the run of P can only send 13 messages to replicas in $\mathcal{C}_2$ if some replicas in $\mathcal{C}_2$ will receive several messages. Neither P nor the replicas in $\mathcal{C}_1$ know which replicas in $\mathcal{C}_2$ have crashed. Hence, in the worst case, the $\mathbf{f}_{\mathcal{C}_2} = 2$ replicas in $\mathcal{C}_2$ that received the most messages have crashed. As we are sending 13 messages and $\mathbf{n}_{\mathcal{C}_2} = 5$, the two replicas that received the most messages must have received at least 6 messages in total. Hence, out of the 13 messages sent, at least 6 can be considered lost. In the run of Figure 4, this loss would happen if $\mathrm{R}_{2,1}$ and $\mathrm{R}_{2,2}$ crash. Consequently, at most $13 - 6 = 7$ messages will arrive at non-faulty replicas. These messages are sent by at most 7 distinct replicas. As $\mathbf{f}_{\mathcal{C}_1} = 7$, all these sending replicas could have crashed. In the run of Figure 4, this loss would happen if $\mathrm{R}_{1,3}$, $\mathrm{R}_{1,4}$, $\mathrm{R}_{1,5}$, $\mathrm{R}_{1,8}$, $\mathrm{R}_{1,9}$, $\mathrm{R}_{1,10}$, and $\mathrm{R}_{1,13}$ crash. Hence, we can thwart any run of P that intends to send 13 messages by crashing $\mathbf{f}_{\mathcal{C}_1}$ replicas in $\mathcal{C}_1$ and $\mathbf{f}_{\mathcal{C}_2}$ replicas in $\mathcal{C}_2$. Consequently, none of the messages of the run will be sent and received by non-faulty replicas, assuring that cluster-sending does not happen.

At least $\mathbf{f}_{\mathcal{C}_1} + 1$ replicas in $\mathcal{C}_1$ need to send messages to non-faulty replicas in $\mathcal{C}_2$ to assure that at least a *single* such message is sent by a non-faulty replica in $\mathsf{nf}(\mathcal{C}_1)$ and, hence, is guaranteed to arrive. We combine this with a thorough analysis along the lines of Example 1 to arrive at the following lower bounds:

Theorem 1. *Let $\mathfrak{S}$ be a system with crash failures, let $\mathcal{C}_1, \mathcal{C}_2 \in \mathfrak{S}$, and let $\{i, j\} = \{1, 2\}$ such that $\mathbf{n}_{\mathcal{C}_i} \geq \mathbf{n}_{\mathcal{C}_j}$. Let $q_i = (\mathbf{f}_{\mathcal{C}_i} + 1) \operatorname{div} \mathbf{nf}_{\mathcal{C}_j}$, $r_i = (\mathbf{f}_{\mathcal{C}_i} +$*

$1) \bmod \mathbf{nf}_{\mathcal{C}_j}$, *and* $\sigma_i = q_i\mathbf{n}_{\mathcal{C}_j} + r_i + \mathbf{f}_{\mathcal{C}_j} \operatorname{sgn} r_i$. *Any protocol that solves the cluster-sending problem in which* $\mathcal{C}_1$ *sends a value* v *to* $\mathcal{C}_2$ *needs to exchange at least* σ_i *messages.*[3]

Proof. The proof uses the same reasoning as Example 1: if a protocol sends at most $\sigma_i - 1$ messages, then we can choose $\mathbf{f}_{\mathcal{C}_1}$ replicas in $\mathcal{C}_1$ and $\mathbf{f}_{\mathcal{C}_2}$ replicas in $\mathcal{C}_2$ that will crash and thus assure that each of the $\sigma_i - 1$ messages is either sent by a crashed replica in $\mathcal{C}_1$ or received by a crashed replica in $\mathcal{C}_2$.

We assume $i = 1$, $j = 2$, and $\mathbf{n}_{\mathcal{C}_1} \geq \mathbf{n}_{\mathcal{C}_2}$. The proof is by contradiction. Hence, assume that a protocol `P` can solve the cluster-sending problem using at most $\sigma_1 - 1$ messages. Consider a run of `P` that sends messages M. Without loss of generality, we can assume that $|M| = \sigma_1 - 1$. Let R be the top $\mathbf{f}_{\mathcal{C}_2}$ receivers of messages in M, let $S = \mathcal{C}_2 \setminus R$, let $M_R \subset M$ be the messages received by replicas in R, and let $N = M \setminus M_R$. We notice that $\mathbf{n}_R = \mathbf{f}_{\mathcal{C}_2}$ and $\mathbf{n}_S = \mathbf{nf}_{\mathcal{C}_2}$.

First, we prove that $|M_R| \geq q_1\mathbf{f}_{\mathcal{C}_2} + \mathbf{f}_{\mathcal{C}_2} \operatorname{sgn} r_1$, this by contradiction. Assume $|M_R| = q_1\mathbf{f}_{\mathcal{C}_2} + \mathbf{f}_{\mathcal{C}_2} \operatorname{sgn} r_1 - v$, $v \geq 1$. Hence, we must have $|N| = q_1\mathbf{nf}_{\mathcal{C}_2} + r_1 + v - 1$. Based on the value r_1, we distinguish two cases. The first case is $r_1 = 0$. In this case, $|M_R| = q_1\mathbf{f}_{\mathcal{C}_2} - v < q_1\mathbf{f}_{\mathcal{C}_2}$ and $|N| = q_1\mathbf{nf}_{\mathcal{C}_2} + v - 1 \geq q_1\mathbf{nf}_{\mathcal{C}_2}$. As $q_1\mathbf{f}_{\mathcal{C}_2} > |M_R|$, there must be a replica in R that received at most $q_1 - 1$ messages. As $|N| \geq q_1\mathbf{nf}_{\mathcal{C}_2}$, there must be a replica in S that received at least q_1 messages. The other case is $r_1 > 0$. In this case, $|M_R| = q_1\mathbf{f}_{\mathcal{C}_2} + \mathbf{f}_{\mathcal{C}_2} - v < (q_1 + 1)\mathbf{f}_{\mathcal{C}_2}$ and $|N| = q_1\mathbf{nf}_{\mathcal{C}_2} + r_1 + v - 1 > q_1\mathbf{nf}_{\mathcal{C}_2}$. As $(q_1 + 1)\mathbf{f}_{\mathcal{C}_2} > |M_R|$, there must be a replica in R that received at most q_1 messages. As $|N| > q_1\mathbf{nf}_{\mathcal{C}_2}$, there must be a replica in S that received at least $q_1 + 1$ messages. In both cases, we identified a replica in S that received more messages than a replica in R, a contradiction. Hence, we must conclude that $|M_R| \geq q_1\mathbf{f}_{\mathcal{C}_2} + \mathbf{f}_{\mathcal{C}_2} \operatorname{sgn} r_1$ and, consequently, $|N| \leq q_1\mathbf{nf}_{\mathcal{C}_2} + r_1 - 1 \leq \mathbf{f}_{\mathcal{C}_1}$. As $\mathbf{n}_R = \mathbf{f}_{\mathcal{C}_2}$, all replicas in R could have crashed, in which case only the messages in N are actually received. As $|N| \leq \mathbf{f}_{\mathcal{C}_1}$, all messages in N could be sent by replicas that have crashed. Hence, in the worst case, no message in M is successfully sent by a non-faulty replica in $\mathcal{C}_1$ and received by a non-faulty replica in $\mathcal{C}_2$, implying that `P` fails. □

The above lower bounds guarantee that at least one message can be delivered. Next, we look at systems with Byzantine failures and replica signing. In this case, at least $2\mathbf{f}_{\mathcal{C}_1} + 1$ replicas in $\mathcal{C}_1$ need to send a replica certificate to non-faulty replicas in $\mathcal{C}_2$ to assure that at least $\mathbf{f}_{\mathcal{C}_1} + 1$ such certificates are sent by non-faulty replicas and, hence, are guaranteed to arrive. Via a similar analysis to the one of Theorem 1, we arrive at:

Theorem 2. *Let* $\mathfrak{S}$ *be a system with Byzantine failures and replica signing, let* $\mathcal{C}_1, \mathcal{C}_2 \in \mathfrak{S}$, *and let* $\{i, j\} = \{1, 2\}$ *such that* $\mathbf{n}_{\mathcal{C}_i} \geq \mathbf{n}_{\mathcal{C}_j}$. *Let* $q_1 = (2\mathbf{f}_{\mathcal{C}_1} +$

[3] Example 1 showed that the impact of faulty replicas is minimal if we minimize the number of messages each replica exchanges. Let $\mathbf{n}_{\mathcal{C}_1} > \mathbf{n}_{\mathcal{C}_2}$. If the number of messages sent to $\mathbf{n}_{\mathcal{C}_2}$ is not a multiple of $\mathbf{n}_{\mathcal{C}_2}$, then minimizing the number of messages received by each replica in $\mathbf{n}_{\mathcal{C}_2}$ means that some replicas in $\mathbf{n}_{\mathcal{C}_2}$ will receive *one* more message than others: each replica in $\mathbf{n}_{\mathcal{C}_2}$ will receive at least q_1 messages, while the term $r_1 + \mathbf{f}_{\mathcal{C}_2} \operatorname{sgn} r_1$ specifies the number of replicas in $\mathbf{n}_{\mathcal{C}_2}$ that will receive $q_1 + 1$ messages.

1) $\operatorname{div} \mathbf{nf}_{\mathcal{C}_2}$, $r_1 = (2\mathbf{f}_{\mathcal{C}_1} + 1) \bmod \mathbf{nf}_{\mathcal{C}_2}$, *and* $\tau_1 = q_1\mathbf{n}_{\mathcal{C}_2} + r_1 + \mathbf{f}_{\mathcal{C}_2} \operatorname{sgn} r_1$; *and let* $q_2 = (\mathbf{f}_{\mathcal{C}_2} + 1) \operatorname{div} (\mathbf{nf}_{\mathcal{C}_1} - \mathbf{f}_{\mathcal{C}_1})$, $r_2 = (\mathbf{f}_{\mathcal{C}_2} + 1) \bmod (\mathbf{nf}_{\mathcal{C}_1} - \mathbf{f}_{\mathcal{C}_1})$, *and* $\tau_2 = q_2\mathbf{n}_{\mathcal{C}_1} + r_2 + 2\mathbf{f}_{\mathcal{C}_1} \operatorname{sgn} r_2$. *Any protocol that solves the cluster-sending problem in which* $\mathcal{C}_1$ *sends a value* v *to* $\mathcal{C}_2$ *needs to exchange at least* τ_i *messages.*[4]

Proof. For simplicity, we assume that each certificate is sent to $\mathcal{C}_2$ in an individual message independent of the other certificates. Hence, each certificate has a sender and a signer (both replicas in $\mathcal{C}_1$) and a receiver (a replica in $\mathcal{C}_2$).

First, we prove the case for $\mathbf{n}_{\mathcal{C}_1} \geq \mathbf{n}_{\mathcal{C}_2}$ using contradiction. Assume that a protocol `P` can solve the cluster-sending problem using at most $\tau_1 - 1$ certificates. Consider a run of `P` that sends messages C, each message representing a single certificate, with $|C| = \tau_1 - 1$. Following the proof of Theorem 1, one can show that, in the worst case, at most $\mathbf{f}_{\mathcal{C}_1}$ messages are sent by non-faulty replicas in $\mathcal{C}_1$ and received by non-faulty replicas in $\mathcal{C}_2$. Now consider the situation in which the faulty replicas in $\mathcal{C}_1$ mimic the behavior in C by sending certificates for another value v' to the same receivers. For the replicas in $\mathcal{C}_2$, the two runs behave the same, as in both cases at most $\mathbf{f}_{\mathcal{C}_1}$ certificates for a value, possibly signed by distinct replicas, are received. Hence, either both runs successfully send values, in which case v' is received by $\mathcal{C}_2$ without agreement, or both runs fail to send values. In both cases, `P` fails to solve the cluster-sending problem.

Next, we prove the case for $\mathbf{n}_{\mathcal{C}_2} \geq \mathbf{n}_{\mathcal{C}_1}$ using contradiction. Assume that a protocol `P` can solve the cluster-sending problem using at most $\tau_2 - 1$ certificates. Consider a run of `P` that sends messages C, each message representing a single certificate, with $|C| = \tau_2 - 1$. Let R be the top $2\mathbf{f}_{\mathcal{C}_1}$ signers of certificates in C, let $C_R \subset C$ be the certificates signed by replicas in R, and let $D = C \setminus C_R$. Via a contradiction argument similar to the one used in the proof of Theorem 1, one can show that $|C_R| \geq 2q_2\mathbf{f}_{\mathcal{C}_1} + 2\mathbf{f}_{\mathcal{C}_1} \operatorname{sgn} r$ and $|D| \leq q_2(\mathbf{nf}_{\mathcal{C}_1} - \mathbf{f}_{\mathcal{C}_1}) + r - 1 = \mathbf{f}_{\mathcal{C}_2}$. As $|D| \leq \mathbf{f}_{\mathcal{C}_2}$, all replicas receiving these certificates could have crashed. Hence, the only certificates that are received by $\mathcal{C}_2$ are in C_R. Partition C_R into two sets of certificates $C_{R,1}$ and $C_{R,2}$ such that both sets contain certificates signed by at most $\mathbf{f}_{\mathcal{C}_1}$ distinct replicas. As the certificates in $C_{R,1}$ and $C_{R,2}$ are signed by $\mathbf{f}_{\mathcal{C}_1}$ distinct replicas, one of these sets can contain only certificates signed by Byzantine replicas. Hence, either $C_{R,1}$ or $C_{R,2}$ could certify a non-agreed upon value v', while only the other set certifies v. Consequently, the replicas in $\mathcal{C}_2$ cannot distinguish between receiving an agreed-upon value v or a non-agreed-upon-value v'. We conclude that `P` fails to solve the cluster-sending problem. □

4 Cluster-Sending via Bijective Sending

In the previous section, we established lower bounds for the cluster-sending problem. Next, we develop *bijective sending*, a powerful technique that allows the design of efficient cluster-sending protocols that match these lower bounds.

[4] Tolerating Byzantine failures in an environment with replica signatures leads to an asymmetry between the sending cluster $\mathcal{C}_1$, in which $2\mathbf{f}_{\mathcal{C}_1} + 1$ replicas need to send, and the receiving cluster $\mathcal{C}_2$, in which only $\mathbf{f}_{\mathcal{C}_2} + 1$ replicas need to receive. This asymmetry results in two distinct cases based on the relative cluster sizes.

Protocol for the sending cluster $\mathcal{C}_1$:

All replicas in $\mathsf{nf}(\mathcal{C}_1)$ agree on v and construct $\langle v\rangle_{\mathcal{C}_1}$.
Choose replicas $S_1 \subseteq \mathcal{C}_1$ and $S_2 \subseteq \mathcal{C}_2$ with $\mathbf{n}_{S_2} = \mathbf{n}_{S_1} = \mathbf{f}_{\mathcal{C}_1} + \mathbf{f}_{\mathcal{C}_2} + 1$.
Choose a bijection $b : S_1 \to S_2$.
for $\textsc{R}_1 \in S_1$ **do**
    $\textsc{R}_1$ sends $(v, \langle v\rangle_{\mathcal{C}_1})$ to $b(\textsc{R}_1)$.

Protocol for the receiving cluster $\mathcal{C}_2$:

event $\textsc{R}_2 \in \mathsf{nf}(\mathcal{C}_2)$ receives $(w, \langle w\rangle_{\mathcal{C}_1})$ from $\textsc{R}_1 \in \mathcal{C}_1$ **do**
    Broadcast $(w, \langle w\rangle_{\mathcal{C}_1})$ to all replicas in $\mathcal{C}_2$.
event $\textsc{R}_2' \in \mathsf{nf}(\mathcal{C}_2)$ receives $(w, \langle w\rangle_{\mathcal{C}_1})$ from $\textsc{R}_2 \in \mathcal{C}_2$ **do**
    $\textsc{R}_2'$ considers w *received*.

Figure 5. `BS-cs`, the bijective sending cluster-sending protocol that sends a value v from $\mathcal{C}_1$ to $\mathcal{C}_2$. We assume Byzantine failures and a system that provides cluster signing.

First, we present a bijective sending protocol for systems with Byzantine failures and cluster signing. Let $\mathcal{C}_1$ be a cluster in which the non-faulty replicas have reached *agreement* on sending a value v to a cluster $\mathcal{C}_2$ and have access to a cluster certificate $\langle v\rangle_{\mathcal{C}_1}$. Let $\mathcal{C}_i$, $i \in \{1, 2\}$, be the cluster with the most replicas. To assure that at least a single non-faulty replica in $\mathcal{C}_1$ sends a message to a non-faulty replica in $\mathcal{C}_2$, we use the lower bound of Theorem 1: we choose σ_i distinct replicas $S_1 \subseteq \mathcal{C}_1$ and replicas $S_2 \subseteq \mathcal{C}_2$ and instruct each replica in $S_1 \subseteq \mathcal{C}_1$ to send v to a distinct replica in $\mathcal{C}_2$. By doing so, we guarantee that at least a single message is sent and received by non-faulty replicas and, hence, guarantee successful cluster-sending. To be able to choose S_1 and S_2 with $\mathbf{n}_{S_1} = \mathbf{n}_{S_2} = \sigma_i$, we need $\sigma_i \leq \min(\mathbf{n}_{\mathcal{C}_1}, \mathbf{n}_{\mathcal{C}_2})$, in which case we have $\sigma_i = \mathbf{f}_{\mathcal{C}_1} + \mathbf{f}_{\mathcal{C}_2} + 1$. The pseudo-code for this *bijective sending* protocol for systems that provide *cluster signing* (`BS-cs`), can be found in Figure 5. Next, we illustrate bijective sending:

Example 2. Consider system $\mathfrak{S} = \{\mathcal{C}_1, \mathcal{C}_2\}$ of Figure 6 with $\mathcal{C}_1 = \{\textsc{R}_{1,1}, \dots, \textsc{R}_{1,8}\}$, $\mathsf{f}(\mathcal{C}_1) = \{\textsc{R}_{1,1}, \textsc{R}_{1,3}, \textsc{R}_{1,4}\}$, $\mathcal{C}_2 = \{\textsc{R}_{2,1}, \dots, \textsc{R}_{2,7}\}$, and $\mathsf{f}(\mathcal{C}_2) = \{\textsc{R}_{2,1}, \textsc{R}_{2,3}\}$. We have $\mathbf{f}_{\mathcal{C}_1} + \mathbf{f}_{\mathcal{C}_2} + 1 = 6$ and we choose $S_1 = \{\textsc{R}_{1,2}, \dots, \textsc{R}_{1,7}\}$, $S_2 = \{\textsc{R}_{2,1}, \dots, \textsc{R}_{2,6}\}$, and $b = \{\textsc{R}_{1,i} \mapsto \textsc{R}_{2,i-1} \mid 2 \leq i \leq 7\}$. Replica $\textsc{R}_{1,2}$ sends a valid message to $\textsc{R}_{2,1}$. As $\textsc{R}_{2,1}$ is faulty, it might ignore this message. Replicas $\textsc{R}_{1,3}$ and $\textsc{R}_{1,4}$ are faulty and might not send a valid message. Additionally, $\textsc{R}_{2,3}$ is faulty and might ignore any message it receives. The messages sent from $\textsc{R}_{1,5}$ to $\textsc{R}_{2,4}$, from $\textsc{R}_{1,6}$ to $\textsc{R}_{2,5}$, and from $\textsc{R}_{1,7}$ to $\textsc{R}_{2,6}$ are all sent by non-faulty replicas to non-faulty replicas. Hence, these messages all arrive correctly.

Having illustrated the concept of bijective sending, as employed by `BS-cs`, we are now ready to prove correctness of `BS-cs`:

Proposition 1. *Let $\mathfrak{S}$ be a system with Byzantine failures and cluster signing and let $\mathcal{C}_1, \mathcal{C}_2 \in \mathfrak{S}$. If $\mathbf{n}_{\mathcal{C}_1} > 2\mathbf{f}_{\mathcal{C}_1}$, $\mathbf{n}_{\mathcal{C}_1} > \mathbf{f}_{\mathcal{C}_1} + \mathbf{f}_{\mathcal{C}_2}$, and $\mathbf{n}_{\mathcal{C}_2} > \mathbf{f}_{\mathcal{C}_1} + \mathbf{f}_{\mathcal{C}_2}$, then* `BS-cs` *satisfies Definition 1 and sends $\mathbf{f}_{\mathcal{C}_1} + \mathbf{f}_{\mathcal{C}_2} + 1$ messages, of size $\mathcal{O}(\|v\|)$ each, between $\mathcal{C}_1$ and $\mathcal{C}_2$.*

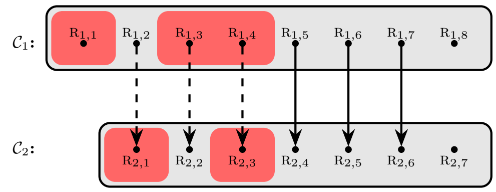

Figure 6. Bijective sending from $\mathcal{C}_1$ to $\mathcal{C}_2$. The faulty replicas are highlighted using a red background. The edges connect replicas $\text{R} \in \mathcal{C}_1$ with $b(\text{R}) \in \mathcal{C}_2$. Each solid edge indicates a message sent and received by non-faulty replicas. Each dashed edge indicates a message sent or received by a faulty replica.

Proof. Choose $S_1 \subseteq \mathcal{C}_1$, $S_2 \subseteq \mathcal{C}_2$, and $b : S_1 \to S_2$ in accordance with `BS-cs` (Figure 5). We have $\mathbf{n}_{S_1} = \mathbf{n}_{S_2} = \mathbf{f}_{\mathcal{C}_1} + \mathbf{f}_{\mathcal{C}_2} + 1$. Let $T = \{b(\text{R}) \mid \text{R} \in \mathsf{nf}(S_1)\}$. By construction, we have $\mathbf{nf}_{S_1} = \mathbf{n}_T \geq \mathbf{f}_{\mathcal{C}_2} + 1$. Hence, we have $\mathbf{nf}_T \geq 1$. Due to Line 5, each replica in $\mathsf{nf}(T)$ will receive the message $(v, \langle v \rangle_{\mathcal{C}_1})$ from a distinct replica in $\mathsf{nf}(S_1)$ and broadcast $(v, \langle v \rangle_{\mathcal{C}_1})$ to all replicas in $\mathcal{C}_2$. As $\mathbf{nf}_T \geq 1$, each replica $\text{R}'_2 \in \mathsf{nf}(\mathcal{C}_2)$ will receive $(v, \langle v \rangle_{\mathcal{C}_1})$ from a replica in $\mathcal{C}_2$ and meet the condition at Line 8, proving *receipt* and *confirmation*. Finally, we have *agreement*, as $\langle v \rangle_{\mathcal{C}_1}$ is non-forgeable. □

To provide cluster-sending in environments with only replica signing, we combine the principle idea of bijective sending with the lower bound on the number of replica certificates exchanged, as provided by Theorem 2. Let $\mathcal{C}_i$, $i \in \{1, 2\}$, be the cluster with the most replicas. To assure that at least $\mathbf{f}_{\mathcal{C}_1} + 1$ non-faulty replicas in $\mathcal{C}_1$ send replica certificates to non-faulty replicas in $\mathcal{C}_2$, we choose sets of replicas $S_1 \subseteq \mathcal{C}_1$ and $S_2 \subseteq \mathcal{C}_2$ with $\mathbf{n}_{S_1} = \mathbf{n}_{S_2} = \tau_i$. To be able to choose S_1 and S_2 with $\mathbf{n}_{S_1} = \mathbf{n}_{S_2} = \tau_i$, we need $\tau_i \leq \min(\mathbf{n}_{\mathcal{C}_1}, \mathbf{n}_{\mathcal{C}_2})$, in which case we have $\tau_i = 2\mathbf{f}_{\mathcal{C}_1} + \mathbf{f}_{\mathcal{C}_2} + 1$. The pseudo-code for this *bijective sending* protocol for systems that provide *replica signing* (`BS-rs`), can be found in Figure 7. Next, we prove the correctness of `BS-rs`:

Proposition 2. *Let $\mathfrak{S}$ be a system with Byzantine failures and replica signing and let $\mathcal{C}_1, \mathcal{C}_2 \in \mathfrak{S}$. If $\mathbf{n}_{\mathcal{C}_1} > 2\mathbf{f}_{\mathcal{C}_1} + \mathbf{f}_{\mathcal{C}_2}$ and $\mathbf{n}_{\mathcal{C}_2} > 2\mathbf{f}_{\mathcal{C}_1} + \mathbf{f}_{\mathcal{C}_2}$, then `BS-rs` satisfies Definition 1 and sends $2\mathbf{f}_{\mathcal{C}_1} + \mathbf{f}_{\mathcal{C}_2} + 1$ messages, of size $\mathcal{O}(\|v\|)$ each, between $\mathcal{C}_1$ and $\mathcal{C}_2$.*

Proof. Choose $S_1 \subseteq \mathcal{C}_1$, $S_2 \subseteq \mathcal{C}_2$, and $b : S_1 \to S_2$ in accordance with `BS-rs` (Figure 7). We have $\mathbf{n}_{S_1} = \mathbf{n}_{S_2} = 2\mathbf{f}_{\mathcal{C}_1} + \mathbf{f}_{\mathcal{C}_2} + 1$. Let $T = \{b(\text{R}) \mid \text{R} \in \mathsf{nf}(S_1)\}$. By construction, we have $\mathbf{nf}_{S_1} = \mathbf{n}_T \geq \mathbf{f}_{\mathcal{C}_1} + \mathbf{f}_{\mathcal{C}_2} + 1$. Hence, we have $\mathbf{nf}_T \geq \mathbf{f}_{\mathcal{C}_1} + 1$. Due to Line 5, each replica in $\mathsf{nf}(T)$ will receive the message $(v, \langle v \rangle_{\text{R}_1})$ from a distinct replica $\text{R}_1 \in \mathsf{nf}(S_1)$ and meet the condition at Line 8, proving *receipt* and *confirmation*.

Next, we prove *agreement*. Consider a value v' not agreed upon by $\mathcal{C}_1$. Hence, no non-faulty replicas $\mathsf{nf}(\mathcal{C}_1)$ will sign v'. Due to non-forgeability of replica certificates, the only certificates that can be constructed for v' are of the form

Protocol for the sending cluster $\mathcal{C}_1$:

1: All replicas in $\mathsf{nf}(\mathcal{C}_1)$ agree on v.
2: Choose replicas $S_1 \subseteq \mathcal{C}_1$ and $S_2 \subseteq \mathcal{C}_2$ with $\mathbf{n}_{S_2} = \mathbf{n}_{S_1} = 2\mathbf{f}_{\mathcal{C}_1} + \mathbf{f}_{\mathcal{C}_2} + 1$.
3: Choose bijection $b : S_1 \to S_2$.
4: **for** $\textsc{r}_1 \in S_1$ **do**
5: $\quad \textsc{r}_1$ sends $(v, \langle v \rangle_{\textsc{r}_1})$ to $b(\textsc{r}_1)$.

Protocol for the receiving cluster $\mathcal{C}_2$:

6: **event** $\textsc{r}_2 \in \mathsf{nf}(\mathcal{C}_2)$ receives $(w, \langle w \rangle_{\textsc{r}_1'})$ from $\textsc{r}_1' \in \mathcal{C}_1$ **do**
7: $\quad$ Broadcast $(w, \langle w \rangle_{\textsc{r}_1'})$ to all replicas in $\mathcal{C}_2$.
8: **event** $\textsc{r}_2' \in \mathsf{nf}(\mathcal{C}_2)$ receives $\mathbf{f}_{\mathcal{C}_1} + 1$ messages $(w, \langle w \rangle_{\textsc{r}_1'})$:
(i) each message is sent by a replica in $\mathcal{C}_2$;
(ii) each message carries the same value w; and
(iii) each message has a distinct signature $\langle w \rangle_{\textsc{r}_1'}$, $\textsc{r}_1' \in \mathcal{C}_1$
do
9: $\quad \textsc{r}_2'$ considers w *received*.

Figure 7. `BS-rs`, the bijective sending cluster-sending protocol that sends a value v from $\mathcal{C}_1$ to $\mathcal{C}_2$. We assume Byzantine failures and a system that provides replica signing.

$\langle v' \rangle_{\textsc{r}_1}$, $\textsc{r}_1 \in \mathsf{f}(\mathcal{C}_1)$. Consequently, each replica in $\mathcal{C}_2$ can only receive and broadcast up to $\mathbf{f}_{\mathcal{C}_1}$ distinct messages of the form $(v', \langle v' \rangle_{\textsc{r}_1'})$, $\textsc{r}_1' \in \mathcal{C}_1$. We conclude that no non-faulty replica will meet the conditions for v' at Line 8. $\square$

5 Cluster-Sending via Partitioning

Unfortunately, the worst-case optimal bijective sending techniques introduced in the previous section are limited to similar-sized clusters:

Example 3. Consider a system $\mathfrak{S}$ with Byzantine failures and cluster certificates. The cluster $\mathcal{C}_1 \in \mathfrak{S}$ wants to send value v to $\mathcal{C}_2 \in \mathfrak{S}$ with $\mathbf{n}_{\mathcal{C}_1} \geq \mathbf{n}_{\mathcal{C}_2}$. To do so, `BS-cs` requires $\sigma_1 = \mathbf{f}_{\mathcal{C}_1} + \mathbf{f}_{\mathcal{C}_2} \leq \mathbf{n}_{\mathcal{C}_2}$. Hence, `BS-cs` requires that $\mathbf{f}_{\mathcal{C}_1}$ is upper-bounded by $\mathbf{nf}_{\mathcal{C}_2} \leq \mathbf{n}_{\mathcal{C}_2}$, which is independent of the size of cluster $\mathcal{C}_1$.

Next, we show how to generalize bijective sending to arbitrary-sized clusters. We do so by *partitioning* the larger-sized cluster into a set of smaller clusters, and then letting sufficient of these smaller clusters participate independently in bijective sending. First, we introduce the relevant partitioning notation.

Definition 2. *Let $\mathfrak{S}$ be a system, let $\mathcal{P}$ be a subset of the replicas in $\mathfrak{S}$, let $c > 0$ be a constant, let $q = \mathbf{n}_{\mathcal{P}} \operatorname{div} c$, and let $r = \mathbf{n}_{\mathcal{P}} \bmod c$. A c-*partition *$\mathsf{partition}(\mathcal{P}) = \{P_1, \dots, P_q, P'\}$ of $\mathcal{P}$ is a partition of the set of replicas $\mathcal{P}$ into sets $P_1, \dots, P_q, P'$ such that $\mathbf{n}_{P_i} = c$, $1 \leq i \leq q$, and $\mathbf{n}_{P'} = r$.*

Example 4. Consider system $\mathfrak{S} = \{\mathcal{C}\}$ of Figure 8 with $\mathcal{C} = \{\textsc{r}_1, \dots, \textsc{r}_{11}\}$ and $\mathsf{f}(\mathcal{C}) = \{\textsc{r}_1, \dots, \textsc{r}_5\}$. The set $\mathsf{partition}(\mathcal{C}) = \{P_1, P_2, P'\}$ with $P_1 = \{\textsc{r}_1, \dots, \textsc{r}_4\}$, $P_2 = \{\textsc{r}_5, \dots, \textsc{r}_8\}$, and $P' = \{\textsc{r}_9, \textsc{r}_{10}, \textsc{r}_{11}\}$ is a 4-partition of $\mathcal{C}$. We have $\mathsf{f}(P_1) =$

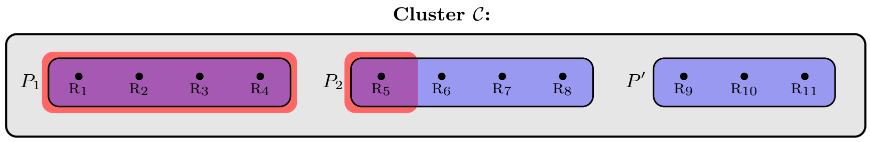

Figure 8. An example of a 4-partition of a cluster $\mathcal{C}$ with 11 replicas, of which the first five are faulty. The three partitions are grouped in blue boxes, the faulty replicas are highlighted using a red background.

P_1, $\mathsf{nf}(P_1) = \emptyset$, and $\mathbf{n}_{P_1} = \mathbf{f}_{P_1} = 4$. Likewise, we have $\mathsf{f}(P_2) = \{R_5\}$, $\mathsf{nf}(P_2) = \{R_6, R_7, R_8\}$, $\mathbf{n}_{P_2} = 4$, and $\mathbf{f}_{P_2} = 1$.

Next, we apply partitioning to `BS-cs`. Let $\mathcal{C}_1$ be a cluster in which the non-faulty replicas have reached agreement on sending a value v to a cluster $\mathcal{C}_2$ and constructed $\langle v \rangle_{\mathcal{C}_1}$. First, we consider the case $\mathbf{n}_{\mathcal{C}_1} \geq \mathbf{n}_{\mathcal{C}_2}$. In this case, we choose a set $P \subseteq \mathcal{C}_1$ of σ_1 replicas in $\mathcal{C}_1$ to sent v to replicas in $\mathcal{C}_2$. To minimize the number of values v received by faulty replicas in $\mathcal{C}_2$, we minimize the number of values v sent to each replica in $\mathcal{C}_2$. Conceptually, we do so by constructing an $\mathbf{n}_{\mathcal{C}_2}$-partition of the σ_1 replicas in P and instruct each resultant set in the partition to perform bijective sending. The pseudo-code for the resultant *sender-partitioned bijective sending* protocol for systems that provide cluster signing, named `SPBS-(`σ_1`,cs)`, can be found in Figure 10. In a similar fashion, we can apply partitioning to `BS-rs`, in which case we instruct τ_1 replicas in $\mathcal{C}_1$ to send v to replicas in $\mathcal{C}_2$, which yields the *sender-partitioned bijective sending* protocol `SPBS-(`τ_1`,rs)` for systems that provide replica signing. Next, we illustrate sender-partitioned bijective sending:

Example 5. We continue from Example 1. Hence, we have $\mathcal{C}_1 = \{R_{1,1}, \dots, R_{1,15}\}$ and $\mathcal{C}_2 = \{R_{2,1}, \dots, R_{2,5}\}$ with $\mathsf{f}(\mathcal{C}_1) = \{R_{1,3}, R_{1,4}, R_{1,5}, R_{1,8}, R_{1,9}, R_{1,10}, R_{1,13}\}$ and $\mathsf{f}(\mathcal{C}_2) = \{R_{2,1}, R_{2,2}\}$. We assume that $\mathfrak{S}$ provides cluster signing and we apply sender-partitioned bijective sending. We have $\mathbf{n}_{\mathcal{C}_1} > \mathbf{n}_{\mathcal{C}_2}$, $q_1 = 8 \operatorname{div} 3 = 2$, $r_1 = 8 \bmod 3 = 2$, and $\sigma_1 = 2 \cdot 5 + 2 + 2 = 14$. We choose the replicas $\mathcal{P} = \{R_{1,1}, \dots, R_{1,14}\} \subseteq \mathcal{C}_1$ and the $\mathbf{n}_{\mathcal{C}_2}$-partition $\mathsf{partition}(\mathcal{P}) = \{P_1, P_2, P'\}$ with $P_1 = \{R_{1,1}, R_{1,2}, R_{1,3}, R_{1,4}, R_{1,5}\}$, $P_2 = \{R_{1,6}, R_{1,7}, R_{1,8}, R_{1,9}, R_{1,10}\}$, and $P' = \{R_{1,11}, R_{1,12}, R_{1,13}, R_{1,14}\}$. Hence, `SPBS-(`$\sigma_1$`,cs)` will perform three rounds of bijective sending. In the first two rounds, `SPBS-(`σ_1`,cs)` will send to all replicas in $\mathcal{C}_2$. In the last round, `SPBS-(`σ_1`,cs)` will send to the replicas $Q = \{R_{2,1}, R_{2,2}, R_{2,3}, R_{2,4}\}$. We choose bijections $b_1 = \{R_{1,1} \mapsto R_{2,1}, \dots, R_{1,5} \mapsto R_{2,5}\}$, $b_2 = \{R_{1,6} \mapsto R_{2,1}, \dots, R_{1,10} \mapsto R_{2,5}\}$, and $b' = \{R_{1,11} \mapsto R_{2,1}, \dots, R_{1,14} \mapsto R_{2,4}\}$. In the first two rounds, we have $\mathbf{f}_{P_1} + \mathbf{f}_{\mathcal{C}_2} = \mathbf{f}_{P_2} + \mathbf{f}_{\mathcal{C}_2} = 3 + 2 = 5 = \mathbf{n}_{\mathcal{C}_2}$. Due to the particular choice of bijections b_1 and b_2, these rounds will fail cluster-sending. In the last round, we have $\mathbf{f}_{P'} + \mathbf{f}_Q = 1 + 2 = 3 < \mathbf{n}_{P'} = \mathbf{n}_Q$. Hence, these two sets of replicas satisfy the conditions of `BS-cs`, can successfully apply bijective sending, and we will have successful cluster-sending (as the non-faulty replica $R_{1,14} \in \mathcal{C}_1$ will send v to the non-faulty replica $R_{2,4} \in \mathcal{C}_2$). We have illustrated the described working of `SPBS-(`σ_1`,cs)` in Figure 9.

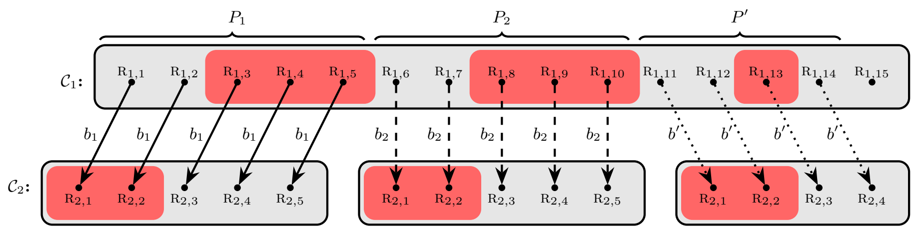

Figure 9. An example of $\texttt{SPBS-(}\sigma_1\texttt{,cs)}$ with $\sigma_1 = 14$ and $\mathsf{partition}(\mathcal{P}) = \{P_1, P_2, P'\}$. Notice that only the instance of bijective sending with the replicas in P' and bijection b' will succeed in cluster-sending.

Protocol for the sending cluster $\mathcal{C}_1$:

The agreement step of $\texttt{BS-}\zeta$ for value v.
Choose replicas $\mathcal{P} \subseteq \mathcal{C}_1$ with $\mathbf{n}_{\mathcal{P}} = \alpha$ and choose $\mathbf{n}_{\mathcal{C}_2}$-partition $\mathsf{partition}(\mathcal{P})$ of $\mathcal{P}$.
for $P \in \mathsf{partition}(\mathcal{P})$ **do**
  Choose replicas $Q \subseteq \mathcal{C}_2$ with $\mathbf{n}_Q = \mathbf{n}_P$ and choose bijection $b : P \rightarrow Q$.
  for $\mathrm{R}_1 \in P$ **do**
    Send v from R_1 to $b(\mathrm{R}_1)$ via the send step of $\texttt{BS-}\zeta$.

Protocol for the receiving cluster $\mathcal{C}_2$:

See the protocol for the receiving cluster in $\texttt{BS-}\zeta$.

Figure 10. $\texttt{SPBS-(}\alpha\texttt{,}\zeta\texttt{)}$, $\zeta \in \{\texttt{cs}, \texttt{rs}\}$, the sender-partitioned bijective sending cluster-sending protocol that sends a value v from $\mathcal{C}_1$ to $\mathcal{C}_2$. We assume the same system properties as $\texttt{BS-}\zeta$.

Next, we prove the correctness of sender-partitioned bijective sending:

Proposition 3. *Let $\mathfrak{S}$ be a system with Byzantine failures, let $\mathcal{C}_1, \mathcal{C}_2 \in \mathfrak{S}$, let σ_1 be as defined in Theorem 1, and let τ_1 be as defined in Theorem 2.*

1. *If $\mathfrak{S}$ provides cluster signing and $\sigma_1 \leq \mathbf{n}_{\mathcal{C}_1}$, then $\texttt{SPBS-(}\sigma_1\texttt{,cs)}$ satisfies Definition 1 and sends σ_1 messages, of size $\mathcal{O}(\|v\|)$ each, between $\mathcal{C}_1$ and $\mathcal{C}_2$.*
2. *If $\mathfrak{S}$ provides replica signing and $\tau_1 \leq \mathbf{n}_{\mathcal{C}_1}$, then $\texttt{SPBS-(}\tau_1\texttt{,rs)}$ satisfies Definition 1 and sends τ_1 messages, of size $\mathcal{O}(\|v\|)$ each, between $\mathcal{C}_1$ and $\mathcal{C}_2$.*

Proof. Let $\beta = (\mathbf{f}_{\mathcal{C}_1} + 1)$ in the case of cluster signing and let $\beta = (2\mathbf{f}_{\mathcal{C}_1} + 1)$ in the case of replica signing. Let $q = \beta \operatorname{div} \mathbf{nf}_{\mathcal{C}_2}$ and $r = \beta \bmod \mathbf{nf}_{\mathcal{C}_2}$. We have $\alpha = q\mathbf{n}_{\mathcal{C}_2} + r + \mathbf{f}_{\mathcal{C}_2} \operatorname{sgn} r$. Choose $\mathcal{P}$ and choose $\mathsf{partition}(\mathcal{P}) = \{P_1, \ldots, P_q, P'\}$ in accordance with $\texttt{SPBS-(}\alpha\texttt{,}\zeta\texttt{)}$ (Figure 10). For each $P \in \mathcal{P}$, choose a Q and b in accordance with $\texttt{SPBS-(}\alpha\texttt{,}\zeta\texttt{)}$, and let $z(P) = \{\mathrm{R} \in P \mid b(\mathrm{R}) \in \mathsf{f}(Q)\}$. As each such b has a distinct domain, the union of them is a surjection $f : \mathcal{P} \rightarrow \mathcal{C}_2$. By construction, we have $\mathbf{n}_{P'} = r + \mathbf{f}_{\mathcal{C}_2} \operatorname{sgn} r$, $\mathbf{n}_{z(P')} \leq \mathbf{f}_{\mathcal{C}_2} \operatorname{sgn} r$, and, for every i, $1 \leq i \leq q$, $\mathbf{n}_{P_i} = \mathbf{n}_{\mathcal{C}_2}$ and $\mathbf{n}_{z(P_i)} = \mathbf{f}_{\mathcal{C}_2}$. Let $V = \mathcal{P} \setminus \left(\bigcup_{P \in \mathsf{partition}(\mathcal{P})} z(P)\right)$. We

Protocol for the sending cluster $\mathcal{C}_1$:

The agreement step of BS-ζ for value v.
Choose replicas $\mathcal{P} \subseteq \mathcal{C}_2$ with $\mathbf{n}_{\mathcal{P}} = \alpha$ and choose $\mathbf{n}_{\mathcal{C}_1}$-partition $\mathsf{partition}(\mathcal{P})$ of $\mathcal{P}$.
for $P \in \mathsf{partition}(\mathcal{P})$ **do**
    Choose replicas $Q \subseteq \mathcal{C}_1$ with $\mathbf{n}_Q = \mathbf{n}_P$ and choose bijection $b : Q \to P$.
    for $\mathrm{R}_1 \in Q$ **do**
        Send v from R_1 to $b(\mathrm{R}_1)$ via the send step of BS-ζ.

Protocol for the receiving cluster $\mathcal{C}_2$:

See the protocol for the receiving cluster in BS-ζ.

Figure 11. RPBS-(α,ζ), $\zeta \in \{\texttt{cs}, \texttt{rs}\}$, the receiver-partitioned bijective sending cluster-sending protocol that sends a value v from $\mathcal{C}_1$ to $\mathcal{C}_2$. We assume the same system properties as BS-ζ.

have

$$\mathbf{n}_V \geq \mathbf{n}_{\mathcal{P}} - (q\mathbf{f}_{\mathcal{C}_2} + \mathbf{f}_{\mathcal{C}_2} \operatorname{sgn} r) = (q\mathbf{n}_{\mathcal{C}_2} + r + \mathbf{f}_{\mathcal{C}_2} \operatorname{sgn} r) - (q\mathbf{f}_{\mathcal{C}_2} + \mathbf{f}_{\mathcal{C}_2} \operatorname{sgn} r) = q\mathbf{nf}_{\mathcal{C}_2} + r = \beta.$$

Let $T = \{f(\mathrm{R}) \mid \mathrm{R} \in \mathsf{nf}(V)\}$. By construction, we have $\mathbf{nf}_T = \mathbf{n}_T$. To complete the proof, we consider cluster signing and replica signing separately. First, the case for cluster signing. As $\mathbf{n}_V \geq \beta = \mathbf{f}_{\mathcal{C}_1} + 1$, we have $\mathbf{nf}_V \geq 1$. By construction, the replicas in $\mathsf{nf}(T)$ will receive the messages $(v, \langle v \rangle_{\mathcal{C}_1})$ from the replicas $\mathrm{R}_1 \in \mathsf{nf}(V)$. Hence, analogous to the proof of Proposition 1, we can prove *receipt*, *confirmation*, and *agreement*. Finally, the case for replica signing. As $\mathbf{n}_V \geq \beta = 2\mathbf{f}_{\mathcal{C}_1} + 1$, we have $\mathbf{nf}_V \geq \mathbf{f}_{\mathcal{C}_1} + 1$. By construction, the replicas in $\mathsf{nf}(T)$ will receive the messages $(v, \langle v \rangle_{\mathrm{R}_1})$ from each replica $\mathrm{R}_1 \in \mathsf{nf}(V)$. Hence, analogous to the proof of Proposition 2, we can prove *receipt*, *confirmation*, and *agreement*. □

Finally, we consider the case $\mathbf{n}_{\mathcal{C}_1} \leq \mathbf{n}_{\mathcal{C}_2}$. In this case, we apply partitioning to BS-cs by choosing a set P of σ_2 replicas in $\mathcal{C}_2$, constructing an $\mathbf{n}_{\mathcal{C}_1}$-partition of P, and instruct $\mathcal{C}_1$ to perform bijective sending with each set in the partition. The pseudo-code for the resultant *receiver-partitioned bijective sending* protocol for systems that provide cluster signing, named RPBS-(σ_2,cs), can be found in Figure 11. In a similar fashion, we can apply partitioning to BS-rs, which yields the *receiver-partitioned bijective sending* protocol RPBS-(τ_2,rs) for systems that provide replica signing. Next, we prove the correctness of these instances of receiver-partitioned bijective sending:

Proposition 4. *Let $\mathfrak{S}$ be a system with Byzantine failures, let $\mathcal{C}_1, \mathcal{C}_2 \in \mathfrak{S}$, let σ_2 be as defined in Theorem 1, and let τ_2 be as defined in Theorem 2.*

1. *If $\mathfrak{S}$ provides cluster signing and $\sigma_2 \leq \mathbf{n}_{\mathcal{C}_2}$, then* RPBS-($\sigma_2$,cs) *satisfies Definition 1 and sends σ_2 messages, of size $\mathcal{O}(\|v\|)$ each, between $\mathcal{C}_1$ and $\mathcal{C}_2$.*
2. *If $\mathfrak{S}$ provides replica signing and $\tau_2 \leq \mathbf{n}_{\mathcal{C}_2}$, then* RPBS-($\tau_2$,rs) *satisfies Definition 1 and sends τ_2 messages, of size $\mathcal{O}(\|v\|)$ each, between $\mathcal{C}_1$ and $\mathcal{C}_2$.*

Proof. Let $\beta = \mathbf{nf}_{\mathcal{C}_1}$ and $\gamma = 1$ in the case of cluster signing and let $\beta = (\mathbf{nf}_{\mathcal{C}_1} - \mathbf{f}_{\mathcal{C}_1})$ and $\gamma = 2$ in the case of replica signing. Let $q = (\mathbf{f}_{\mathcal{C}_2} + 1) \operatorname{div} \beta$ and $r = (\mathbf{f}_{\mathcal{C}_2} + 1) \bmod \beta$. We have $\alpha = q\mathbf{n}_{\mathcal{C}_1} + r + \gamma \mathbf{f}_{\mathcal{C}_1} \operatorname{sgn} r$. Choose $\mathcal{P}$ and choose $\mathsf{partition}(\mathcal{P}) = \{P_1, \dots, P_q, P'\}$ in accordance with $\texttt{RPBS-(}\alpha\texttt{,}\zeta\texttt{)}$ (Figure 11). For each $P \in \mathcal{P}$, choose a Q and b in accordance with $\texttt{RPBS-(}\alpha\texttt{,}\zeta\texttt{)}$, and let $z(P) = \{\mathrm{R} \in P \mid b^{-1}(\mathrm{R}) \in \mathsf{f}(Q)\}$. As each such b^{-1} has a distinct domain, the union of them is a surjection $f^{-1} : \mathcal{P} \to \mathcal{C}_1$. By construction, we have $\mathbf{n}_{P'} = r + \gamma \mathbf{f}_{\mathcal{C}_1} \operatorname{sgn} r$, $\mathbf{n}_{z(P')} \leq \mathbf{f}_{\mathcal{C}_1} \operatorname{sgn} r$, and, for every i, $1 \leq i \leq q$, $\mathbf{n}_{P_i} = \mathbf{n}_{\mathcal{C}_1}$ and $\mathbf{n}_{z(P_i)} = \mathbf{f}_{\mathcal{C}_1}$. Let $T = \mathcal{P} \setminus \left(\bigcup_{P \in \mathsf{partition}(\mathcal{P})} z(P)\right)$. We have

$$\begin{aligned}\mathbf{n}_T &\geq \mathbf{n}_{\mathcal{P}} - (q\mathbf{f}_{\mathcal{C}_1} + \mathbf{f}_{\mathcal{C}_1} \operatorname{sgn} r) = (q\mathbf{n}_{\mathcal{C}_1} + r + \gamma \mathbf{f}_{\mathcal{C}_1} \operatorname{sgn} r) - (q\mathbf{f}_{\mathcal{C}_1} + \mathbf{f}_{\mathcal{C}_1} \operatorname{sgn} r)\\ &= q\mathbf{nf}_{\mathcal{C}_1} + r + (\gamma - 1)\mathbf{f}_{\mathcal{C}_1} \operatorname{sgn} r.\end{aligned}$$

To complete the proof, we consider cluster signing and replica signing separately.

First, the case for cluster signing. We have $\beta = \mathbf{nf}_{\mathcal{C}_1}$ and $\gamma = 1$. Hence, $\mathbf{n}_T \geq q\mathbf{nf}_{\mathcal{C}_1} + r + (\gamma - 1)\mathbf{f}_{\mathcal{C}_1} \operatorname{sgn} r = q\beta + r = \mathbf{f}_{\mathcal{C}_2} + 1$. We have $\mathbf{nf}_T \geq \mathbf{n}_T - \mathbf{f}_{\mathcal{C}_2} \geq 1$. Let $V = \{f^{-1}(\mathrm{R}) \mid \mathrm{R} \in \mathsf{nf}(T)\}$. By construction, we have $\mathbf{nf}_V = \mathbf{n}_V$ and we have $\mathbf{nf}_V \geq 1$. Consequently, the replicas in $\mathsf{nf}(T)$ will receive the messages $(v, \langle v \rangle_{\mathcal{C}_1})$ from the replicas $\mathrm{R}_1 \in \mathsf{nf}(V)$. Analogous to the proof of Proposition 1, we can prove *receipt*, *confirmation*, and *agreement*.

Finally, the case for replica signing. We have $\beta = \mathbf{nf}_{\mathcal{C}_1} - \mathbf{f}_{\mathcal{C}_1}$ and $\gamma = 2$. Hence, $\mathbf{n}_T \geq q\mathbf{nf}_{\mathcal{C}_1} + r + (\gamma - 1)\mathbf{f}_{\mathcal{C}_1} \operatorname{sgn} r = q(\beta + \mathbf{f}_{\mathcal{C}_1}) + r + \mathbf{f}_{\mathcal{C}_1} \operatorname{sgn} r = (q\beta + r) + q\mathbf{f}_{\mathcal{C}_1} + \mathbf{f}_{\mathcal{C}_1} \operatorname{sgn} r = (\mathbf{f}_{\mathcal{C}_2} + 1) + q\mathbf{f}_{\mathcal{C}_1} + \mathbf{f}_{\mathcal{C}_1} \operatorname{sgn} r$. We have $\mathbf{nf}_T \geq q\mathbf{f}_{\mathcal{C}_1} + \mathbf{f}_{\mathcal{C}_1} \operatorname{sgn} r + 1 = (q + \operatorname{sgn} r)\mathbf{f}_{\mathcal{C}_1} + 1$. As there are $(q + \operatorname{sgn} r)$ non-empty sets in $\mathsf{partition}(\mathcal{P})$, there must be a set $P \in \mathcal{P}$ with $\mathbf{n}_{P \cap \mathbf{nf}_T} \geq \mathbf{f}_{\mathcal{C}_1} + 1$. Let b be the bijection chosen earlier for P and let $V = \{b^{-1}(\mathrm{R}) \mid \mathrm{R} \in (P \cap \mathbf{nf}_T)\}$. By construction, we have $\mathbf{nf}_V = \mathbf{n}_V$ and we have $\mathbf{nf}_V \geq \mathbf{f}_{\mathcal{C}_1} + 1$. Consequently, the replicas in $\mathsf{nf}(T)$ will receive the messages $(v, \langle v \rangle_{\mathrm{R}_1})$ from each replica $\mathrm{R}_1 \in \mathsf{nf}(V)$. Hence, analogous to the proof of Proposition 2, we can prove *receipt*, *confirmation*, and *agreement*. $\square$

The bijective sending cluster-sending protocols, the sender-partitioned bijective cluster-sending protocols, and the receiver-partitioned bijective cluster-sending protocols each deal with differently-sized clusters. By choosing the applicable protocols, we have the following:

Corollary 1. *Let $\mathfrak{S}$ be a system, let $\mathcal{C}_1, \mathcal{C}_2 \in \mathfrak{S}$, let σ_1 and σ_2 be as defined in Theorem 1, and let τ_1 and τ_2 be as defined in Theorem 2. Consider the cluster-sending problem in which $\mathcal{C}_1$ sends a value v to $\mathcal{C}_2$.*

1. *If $\mathbf{n}_{\mathcal{C}} > 3\mathbf{f}_{\mathcal{C}}$, $\mathcal{C} \in \mathfrak{S}$, and $\mathfrak{S}$ has crash failures, omit failures, or Byzantine failures and cluster signing, then* $\texttt{BS-cs}$, $\texttt{SPBS-(}\sigma_1\texttt{,cs)}$, *and* $\texttt{RPBS-(}\sigma_2\texttt{,cs)}$ *are a solution to the cluster-sending problem with optimal message complexity. These protocols solve the cluster-sending problem using $\mathcal{O}(\max(\mathbf{n}_{\mathcal{C}_1}, \mathbf{n}_{\mathcal{C}_2}))$ messages, of size $\mathcal{O}(\|v\|)$ each.*
2. *If $\mathbf{n}_{\mathcal{C}} > 4\mathbf{f}_{\mathcal{C}}$, $\mathcal{C} \in \mathfrak{S}$, and $\mathfrak{S}$ has Byzantine failures and replica signing, then* $\texttt{BS-rs}$, $\texttt{SPBS-(}\tau_1\texttt{,rs)}$, *and* $\texttt{RPBS-(}\tau_2\texttt{,rs)}$ *are a solution to the cluster-sending problem with optimal replica certificate usage. These protocols solve*

the cluster-sending problem using $\mathcal{O}(\max(\mathbf{n}_{\mathcal{C}_1}, \mathbf{n}_{\mathcal{C}_2}))$ messages, of size $\mathcal{O}(\|v\|)$ each.

6 Performance Evaluation

In the previous sections, we introduced *worst-case optimal* cluster-sending protocols. To gain further insight in the performance attainable by these protocols, we implemented these protocols in a simulated sharded resilient system environment that allows us to control the faulty replicas in each cluster. In the experiments, we used equal-sized clusters, which corresponds to the setup used by recent sharded consensus-based system proposals [1,3,10]. Hence, we used only the bijective cluster-sending protocols `BS-cs` and `BS-rs`. As a baseline of comparison, we also evaluated the *broadcast-based cluster-sending protocol* of `Chainspace` [1] that can perform cluster-sending using $\mathbf{n}_{\mathcal{C}_1} \cdot \mathbf{n}_{\mathcal{C}_2}$ messages. We refer to Figure 2 for a theoretical comparison between our cluster-sending protocols and the protocol utilized by `Chainspace`. Furthermore, we have implemented `MC-cs` and `MC-rs`, two multicast-based cluster-sending protocols, one using cluster signing and the other using replica signing), that work similar to the protocol of `Chainspace`, but minimize the number of messages to provide cluster-sending.

In the experiment, we measured the number of messages exchanged as a function of the number of faulty replicas. In specific, we measured the number of messages exchanged in 10 000 runs of the cluster-sending protocols under consideration. In each run we measure the number of messages exchanged when sending a value v from a cluster $\mathcal{C}_1$ to a cluster $\mathcal{C}_2$ with $\mathbf{n}_{\mathcal{C}_1} = \mathbf{n}_{\mathcal{C}_2} = 3\mathbf{f}_{\mathcal{C}_1} + 1 = 3\mathbf{f}_{\mathcal{C}_2} + 1$, and we aggregate this data over 10 000 runs. Furthermore, we measured in each run the number of messages exchanged between non-faulty replicas, as these messages are necessary to guarantee cluster-sending. The results of the experiment can be found in Figure 12.

As is clear from the results, our worst-case optimal cluster-sending protocols are able to out-perform existing cluster-sending protocols by a wide margin, which is a direct consequence of the difference between quadratic message complexity (`Chainspace`, `MC-cs`, and `MC-rs`) and a worst-case optimal linear message complexity (`BS-cs` and `BS-rs`). As can be seen in Figure 12, *right*, our protocols do so by massively cutting back on sending messages between faulty replicas, while still ensuring that in all cases sufficient messages are exchanged between non-faulty replicas (thereby assuring cluster-sending).

7 Conclusion

In this paper, we identified and formalized the *cluster-sending problem*, a fundamental primitive in the design and implementation of blockchain-inspired sharded fault-tolerant data processing systems. Not only did we formalize the cluster-sending problem, we also proved lower bounds on the complexity of this problem. Furthermore, we developed bijective sending and partitioned bijective sending, two powerful techniques that can be used in the construction of practical

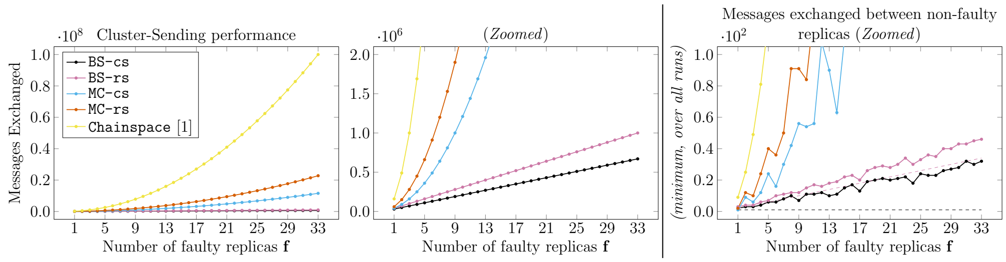

Figure 12. A comparison of the number of message exchange steps as a function of the number of faulty replicas in both clusters by our *worst-case optimal* cluster-sending protocols `BS-cs` and `BS-rs`, and by three protocols based on the literature. For each protocol, we measured the number of message exchange steps to send 10 000 values between two equally-sized clusters, each cluster having $n = 3\mathbf{f} + 1$ replicas. The dashed lines in the plot on the *right* indicate the minimum number of messages that need to be exchanged between non-faulty replicas for the protocols `BS-cs` and `BS-rs`, respectively, to guarantee cluster-sending (no protocol can do better).

cluster-sending protocols with optimal complexity that matches the lower bounds established. We believe that our work provides a strong foundation for future blockchain-inspired sharded fault-tolerant data processing systems that can deal with Byzantine failures and the challenges of large-scale data processing.

Our fundamental results open a number of key research avenues to further high-performance fault-tolerant data processing. First, we are interested in further improving our understanding of cluster-sending, e.g., by establishing lower bounds on cluster-sending in the absence of public-key cryptography and in the absence of reliable networks. Second, we are interested in improved cluster-sending protocols that can perform cluster-sending with less-than a linear number of messages, e.g., by using randomization or by optimizing for cases without failures. Finally, we are interested in putting cluster-sending protocols to practice by incorporating them in the design of practical sharded fault-tolerant systems, thereby moving even closer to general-purpose high-performance fault-tolerant data processing.

References

1. Al-Bassam, M., Sonnino, A., Bano, S., Hrycyszyn, D., Danezis, G.: Chainspace: A sharded smart contracts platform (2017), http://arxiv.org/abs/1708.03778
2. Castro, M., Liskov, B.: Practical byzantine fault tolerance and proactive recovery. ACM Trans. Comput. Syst. **20**(4), 398–461 (2002). https://doi.org/10.1145/571637.571640
3. Dang, H., Dinh, T.T.A., Loghin, D., Chang, E.C., Lin, Q., Ooi, B.C.: Towards scaling blockchain systems via sharding. In: Proceedings of the 2019 International Conference on Management of Data. pp. 123–140. ACM (2019). https://doi.org/10.1145/3299869.3319889
4. Dolev, D., Strong, H.R.: Authenticated algorithms for byzantine agreement. SIAM J. Comput. **12**(4), 656–666 (1983). https://doi.org/10.1137/0212045

5. Gordon, W.J., Catalini, C.: Blockchain technology for healthcare: Facilitating the transition to patient-driven interoperability. Comput. Struct. Biotechnol. J. **16**, 224–230 (2018). https://doi.org/10.1016/j.csbj.2018.06.003
6. Gupta, S., Hellings, J., Sadoghi, M.: Fault-Tolerant Distributed Transactions on Blockchain. Synthesis Lectures on Data Management, Morgan & Claypool (2021). https://doi.org/10.2200/S01068ED1V01Y202012DTM065
7. Gupta, S., Hellings, J., Sadoghi, M.: RCC: Resilient concurrent consensus for high-throughput secure transaction processing. In: 2021 IEEE 37th International Conference on Data Engineering (ICDE). pp. 1392–1403. IEEE (2021). https://doi.org/10.1109/ICDE51399.2021.00124
8. Gupta, S., Rahnama, S., Hellings, J., Sadoghi, M.: ResilientDB: Global scale resilient blockchain fabric. Proc. VLDB Endow. **13**(6), 868–883 (2020). https://doi.org/10.14778/3380750.3380757
9. Gupta, S., Rahnama, S., Hellings, J., Sadoghi, M.: Proof-of-Execution: Reaching consensus through fault-tolerant speculation. In: Proceedings of the 24th International Conference on Extending Database Technology (EDBT). pp. 301–312. OpenProceedings.org (2021). https://doi.org/10.5441/002/edbt.2021.27
10. Hellings, J., Sadoghi, M.: ByShard: Sharding in a byzantine environment. Proc. VLDB Endow. **14**(11), 2230–2243 (2021). https://doi.org/10.14778/3476249.3476275
11. Herlihy, M.: Atomic cross-chain swaps. In: Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing. pp. 245–254. ACM (2018). https://doi.org/10.1145/3212734.3212736
12. Herlihy, M., Liskov, B., Shrira, L.: Cross-chain deals and adversarial commerce. Proc. VLDB Endow. **13**(2), 100–113 (2019). https://doi.org/10.14778/3364324.3364326
13. Kamel Boulos, M.N., Wilson, J.T., Clauson, K.A.: Geospatial blockchain: promises, challenges, and scenarios in health and healthcare. Int. J. Health. Geogr. **17**(1), 1211–1220 (2018). https://doi.org/10.1186/s12942-018-0144-x
14. Lao, L., Li, Z., Hou, S., Xiao, B., Guo, S., Yang, Y.: A survey of IoT applications in blockchain systems: Architecture, consensus, and traffic modeling. ACM Comput. Surv. **53**(1) (2020). https://doi.org/10.1145/3372136
15. Menezes, A.J., Vanstone, S.A., Oorschot, P.C.V.: Handbook of Applied Cryptography. CRC Press, Inc., 1st edn. (1996)
16. Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system, https://bitcoin.org/en/bitcoin-paper
17. Özsu, M.T., Valduriez, P.: Principles of Distributed Database Systems. Springer (2020). https://doi.org/10.1007/978-3-030-26253-2
18. Rejeb, A., Keogh, J.G., Zailani, S., Treiblmaier, H., Rejeb, K.: Blockchain technology in the food industry: A review of potentials, challenges and future research directions. Logistics **4**(4) (2020). https://doi.org/10.3390/logistics4040027
19. Shoup, V.: Practical threshold signatures. In: Advances in Cryptology — EUROCRYPT 2000. pp. 207–220. Springer (2000)
20. Treiblmaier, H., Beck, R. (eds.): Business Transformation through Blockchain. Springer (2019). https://doi.org/10.1007/978-3-319-98911-2
21. Wood, G.: Ethereum: a secure decentralised generalised transaction ledger, https://gavwood.com/paper.pdf, EIP-150 revision
22. Wu, M., Wang, K., Cai, X., Guo, S., Guo, M., Rong, C.: A comprehensive survey of blockchain: From theory to IoT applications and beyond. IEEE Internet Things J. **6**(5), 8114–8154 (2019). https://doi.org/10.1109/JIOT.2019.2922538

23. Yin, M., Malkhi, D., Reiter, M.K., Gueta, G.G., Abraham, I.: HotStuff: BFT consensus with linearity and responsiveness. In: Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing. pp. 347–356. ACM (2019). https://doi.org/10.1145/3293611.3331591
24. Zakhary, V., Agrawal, D., El Abbadi, A.: Atomic commitment across blockchains. Proc. VLDB Endow. **13**(9), 1319–1331 (2020). https://doi.org/10.14778/3397230.3397231