Leveraging Multicast Primitives for Highly Available Replication in Drone Swarms

João Victor Nascimento[1], André Brito[1], Ewerton Salvador[1], Jiayi Chen[2], Tassany Onofre[2], Emmanuel Lochin[2], Guthemberg Silvestre[2]

[1] Centro de Informatica, Universidade Federal da Paraíba (UFPB), João Pessoa, Brazil

[2]Fédération ENAC ISAE-SUPAERO ONERA, Université de Toulouse, France.

{andre.oliveira-brito,joao.silva-bezerra-nascimento}@enac.alumni.fr,

ewerton@ci.ufpb.br, jiayi.chen@recherche.enac.fr,

{tassany.onofre-de-oliveira,emmanuel.lochin,silvestre}@enac.fr

Abstract. *Drone swarms have emerged as promising robotic platforms to solve an increasing number of real-world problems. Yet, the design of fully distributed services atop drones remains challenging largely due to limitations in underlying network stacks. To address this, we make a case for highly available replication as a key building block for distributed computing in the face of ad hoc, lossy wireless networks. Our work leverages multicast primitives and Conflict-free Replicated Data Types (CRDTs) to improve replica synchronisation. Our emulation study indicates that state-of-the-art multicast primitives play an important role on data replication in drone swarms, highlighting a fundamental trade-off between synchronisation efficiency and communication overhead.*

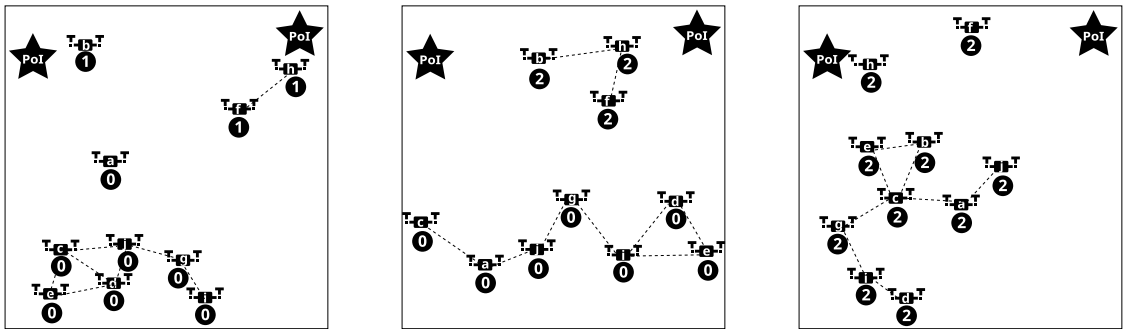
1. Introduction

Drone swarms are becoming increasingly important as platforms for real-world services with great societal impact, such as search and rescue operations [Horyna et al. 2023], environmental hazard monitoring [Yuan et al. 2015, Villa et al. 2016], air quality monitoring [Hossein Motlagh et al. 2025], and ecology and wildlife remote sensing [Ivošević et al. 2015]. Although the size of these platforms may vary according to the service, the ad hoc wireless technology and the coverage area, they commonly comprise a few tens of drones connected through a sparse communication network [Chung et al. 2016, Vásárhelyi et al. 2014, Preiss et al. 2017, Weinstein et al. 2018], mostly due to operational complexity and costs [Spica et al. 2013, Hattenberger et al. 2014].

Towards consolidated architectures, a significant body of research [Schranz et al. 2021] focuses on how to enable collaborative, autonomous functionalities at the swarm level. Unsurprisingly, these functionalities strongly rely on data sharing and continuous communication among drones to function properly, which are both hindered by the mobility of drones and frequent network partitions. To address these issues and to foster the design of distributed computing in drone swarms, we make a case for highly available replication based on Conflict-free Replicated Data Types (CRDTs) [Shapiro et al. 2011]. By trading consistency for distributed computing availability, CRDTs provide a replication technique where operations are executed locally regardless of the state of the network. To synchronise replicas, nodes periodically send updates across the network.

As the updates are processed, CRDTs guarantee strong eventual consistency: replicas will eventually converge to the same state in a deterministic manner.

Figure 1 illustrates three sequential stages of a swarm of 10 drones exploring a squared area to inspect two points of interest (PoI) using CRDTs for fault-tolerant, highly available inspection data replication. In Figure 1a, the network is partitioned into three components. Drones `b` and `h` inspect the two points of interest and update their states locally. Periodically, drones multicast their new states to their neighbourhood, which allows `f` to update its state. As drones move, `b` joins `h` and `f`, and a new network partition with three nodes emerges, as depicted in Figure 1b. Finally, `b` joins the biggest network component; it multicasts its state, which allows the swarm to converge to the same state, as shown in Figure 1c. Despite network partitions, the network stack plays a key role in replica synchronisation: faster synchronisation leads to faster convergence.



(a) Initial operations. **(b) Synchronisation.** **(c) Convergence.**

Figure 1. Regardless of the network partitions, the swarm eventually converges to the same state.

In this work, we assess the impact of different network settings on the synchronisation of CRDT replicas atop drone swarms. To this end, we compare three state-of-the-art multicast primitives, namely best-effort, flooding and gossip multicasting. While the two former primitives leverage the latest version of `batman` [Johnson et al. 2008], a real-world, consolidated IP routing protocol for highly mobile ad hoc networks, the third one implements a gossip-based dissemination protocol, proposed by Drabkin et al. [Drabkin et al. 2007], as a middleware for mobile ad hoc networks that passes messages directly to the link layer. Our preliminary findings suggest that a gossip protocol outperforms `batman`-based counterparts in terms of synchronisation time with superior scalability. However, `batman`-based flooding multicast remains competitive with a lower communication overhead. Overall, this work makes three contributions. First, we propose a highly available replication scheme to foster distributed computing in drone swarms. Second, we provide detailed guidelines to co-design and implement CRDTs with multicast primitives. Third, via extensive evaluations, we demonstrate the efficiency of different replica synchronisation approaches that are tolerant to network partitions.

The remainder of this paper is organised as follows. Section 2 surveys the prior work on swarm robotics and distributed computing, highlighting CRDTs as a promising technique for services in mobile ad hoc networks. We detail in Section 3 the guidelines to co-design and implement CRDTs with communication primitives. Then, in Section 4, we describe the evaluation of replica synchronisation performance in drone swarms and

we explain the main results and findings. Finally, Section 6 concludes.

2. Background

Effective coordination in drone swarms relies heavily on distributed synchronisation mechanisms that allow nodes to exchange state, maintain a consistent view of shared data, and support collective decision making. This requirement is particularly challenging in mobile settings, where communication links are intermittent, network topology changes frequently due to mobility, and available bandwidth and energy resources are limited. Additionally, the *CAP Theorem* [Gilbert and Lynch 2012] states that a distributed system can simultaneously satisfy at most two of the following properties: consistency, availability, and partition tolerance. In highly dynamic swarm networks, network partitions are frequent and largely unavoidable, which makes it difficult for traditional strongly consistent systems to maintain high availability. As a result, such systems are hard to deploy efficiently in drone swarms, where continuous operation and timely coordination are essential.

CRDTs enable replication with strong eventual consistency despite network partitions. CRDTs [Shapiro et al. 2011] are data structures designed to be replicated across multiple nodes, where each replica can be updated independently without coordination. Their core principle is that all updates are constructed to be commutative, associative, and idempotent, ensuring that replicas eventually converge to the same state once all updates have been disseminated, regardless of the order or timing of their delivery. CRDTs offer strong eventual consistency without global coordination, making them the preferred choice for systems that must remain operational during network partitions and synchronise asynchronously, which are common in drone swarms. Current research, such as *Log-Structured CRDTs* [Saquib et al. 2021] and *ConflictSync* [Gomes et al. 2025], focuses heavily on bandwidth efficiency and delta compression. However, these analyses typically assume ideal network abstractions or simplistic transport layers, isolating the data structure performance from the chaotic reality of mobile ad hoc links where channel contention is frequent.

While CRDTs relax the need for strict coordination, their correctness and efficiency still depend on the timely dissemination of updates among nodes. The way updates are propagated through the network has a direct impact on convergence speed, communication overhead, and overall system performance. Dissemination strategies therefore play a key role as a communication support layer for CRDT-based synchronisation in drone swarms. Regarding the communication layer, two major paradigms are commonly considered for mobile swarms.

***Batman* maintains reliable connectivity in highly mobile swarm networks.** The first paradigm is based on routing, and is prominently represented by the Better Approach To Mobile Ad hoc Networking (`batman`) protocol [Johnson et al. 2008]. `batman` is a proactive layer-two routing protocol designed for mobile ad hoc networks, where each node maintains next hop information rather than complete end-to-end paths. By continuously exchanging small originator messages, nodes are able to estimate link quality and select the most reliable neighbor for forwarding traffic. This localized decision process limits routing state complexity and allows `batman` to react efficiently to topology changes caused by mobility. Comparative studies in Flying Ad hoc Network (FANET)

scenarios have evaluated `batman` alongside other routing protocols under high mobility conditions [Guillen-Perez et al. 2021], making it a relevant communication substrate for drone swarm deployments.

Gossip-based dissemination improves scalability by reducing redundant transmissions. An alternative paradigm relies on gossip-based communication. The theoretical foundations of gossip protocols are well established in [Shah 2009], and a number of practical designs have been proposed to adapt these ideas to resource constrained and dynamic networks. Protocols such as RAPID [Drabkin et al. 2007] and Trickle [Levis et al. 2008] implement so-called "polite gossip" mechanisms, in which broadcast activity is regulated according to local utility and network conditions. Rather than blindly flooding messages, nodes exchange lightweight metadata to detect missing information and trigger selective retransmissions when needed. This approach reduces redundant transmissions while preserving a high probability of delivery. More advanced techniques, such as MIXER [Herrmann et al. 2018], further improve dissemination efficiency by introducing network coding into the gossip process.

Despite the richness of this literature, most existing works study gossip-based protocols and routing-based broadcast mechanisms in isolation. There is still limited empirical evidence that directly compares high-level gossip-based dissemination protocols with broadcast-oriented reactive flooding over routing substrates such as `batman`, especially from the perspective of application-level performance metrics. This gap is particularly relevant for CRDT synchronisation in mobile drone swarms, where communication behavior directly influences convergence speed, overhead, and robustness.

Motivated by these observations, this work evaluates the performance of CRDT synchronisation under three representative dissemination strategies. By comparing their behavior in a mobile drone swarm setting, we aim to improve the understanding of how dissemination choices influence the efficiency and practicality of CRDT-based coordination mechanisms in real-world swarm deployments.

3. Designing a highly available replication for drone swarms

In this section, we introduce a general, layered-system architecture to enable highly available replication for drone swarms. First, we briefly present a suitable system model. Based on this model, we motivate the use of CRDTs as a key component to design replication that is tolerant to network partitions. To enable CRDT-based replication, we present a set of guidelines to co-design different layers of the network stack in order to improve the efficiency of replication based on CRDTs. Finally, we provide the implementation details of our architectural recommendations.

3.1. System model

The replication investigated in this work is considered in a drone swarm that consists of a set of N nodes, constituting a replica set, whose goal is to explore an area and to replicate collected data.

Failure model. Drones and sensors are subject to omission failure but do not experience arbitrary behaviour (i.e., no Byzantine failures). The communication network and communication channels are both unreliable, mostly due to packet loss and continuous modifications of the sparse network topology. The network is also subject to unpredictable

latencies, as well as load imbalances (e.g., peak demand), that are imposed on both nodes and the message exchange. Such imbalances may cause variations in transmission delays. Therefore, transmitted messages can be lost but not corrupted. Consequently, the system tolerates f faulty nodes such that $f < N$.

Timing model. We assume that the system is asynchronous; no assumptions about process execution speeds or message delivery times are made.

3.2. Highly available replication with CRDTs

Data replication serves as the backbone for decentralized coordination in drone swarms. Whether sharing exploration maps or task allocation lists, the swarm relies on a consistent shared view of the global mission status to make autonomous decisions. However, the high mobility and limited communication range of drones frequently partition the network, rendering synchronous replication strategies impractical. To ensure high availability under these conditions, we rely on CRDTs.

Unlike consensus-based approaches (e.g., Paxos, Raft) that sacrifice availability to maintain linearizability during partitions, CRDTs guarantee *Strong Eventual Consistency* (SEC) without requiring central coordination. This property is fundamental for our system model, as it allows drones to update their local state independently even when isolated, guaranteeing that all replicas will eventually converge to the same state once connectivity is restored.

We specifically adopt State-based CRDTs (CvRDTs) over Operation-based approaches (CmRDTs). While CmRDTs typically entail lower payload overhead, they need reliable causal broadcast guarantees from the network layer. Sustaining such guarantees in sparse, unreliable ad hoc networks is computationally expensive and prone to stalling. In contrast, CvRDTs are modeled as a join-semilattice where the merge operation is idempotent, commutative, and associative. This design aligns with the unreliable nature of the communication channels found in mobile swarms, making the system inherently resilient to message duplication and out-of-order delivery.

To mitigate the dissemination overhead inherent to standard CvRDTs—where the naive approach propagates the full local state—we leverage the *Delta-based synchronisation model* (δ-CRDTs) proposed by Enes et al. [Enes et al. 2019]. Instead of transmitting the entire lattice, the system generates minimal, incremental state fragments called *deltas* (δ). These deltas encapsulate strictly the recent local modifications, which are then merged into the remote state ($S' = S \sqcup \delta$). This approach significantly reduces bandwidth consumption while preserving the convergence guarantees of the original state-based model.

3.3. Multicast primitives and network-CRDT co-design

Multicast primitives [Janic 2005] are fundamental for information dissemination in networked systems. In this work, these primitives allow drones to synchronise their CRDT replicas. As drones communicate through a sparse, wireless ad hoc network, we assume that state synchronisation messages are exchanged through unreliable channels to reduce synchronisation costs. For this reason, we are interested only in multicast protocols that are lightweight (*i.e.*, no order guarantees) and tolerant to network partitions. Therefore, we select the following three multicast primitives for this study:

Best-effort multicast: is a straightforward manner to disseminate information to drones of a swarm. Essentially it guarantees that if the drone continuously sends unicast messages for synchronisation, enough messages are eventually delivered to all non-faulty drones in a swarm and synchronisation is reached.
Flooding multicast: leverages a pre-configured multicast address to continuously send messages to a specific group of drones in a deterministic way. Each non-faulty drone simply forwards a message m to each of its neighbours, except to the one from which it received m. To avoid network overload, a drone keeps track of the messages it received and forwarded to ignore duplicates.
Gossip multicast: is a simple, reliable probabilistic primitive based only on local information. It exchanges state synchronisation messages interactively; in each step, a drone only communicates with a small subset of the swarm, exchanging a small amount of synchronisation information only. No assumption about the availability of multi-hop routing is made, and it only requires maintaining an approximation of the system membership.

To leverage these multicast primitives for efficient CRDT synchronisation, we propose three cross-layer designs, shown in Figure 2.

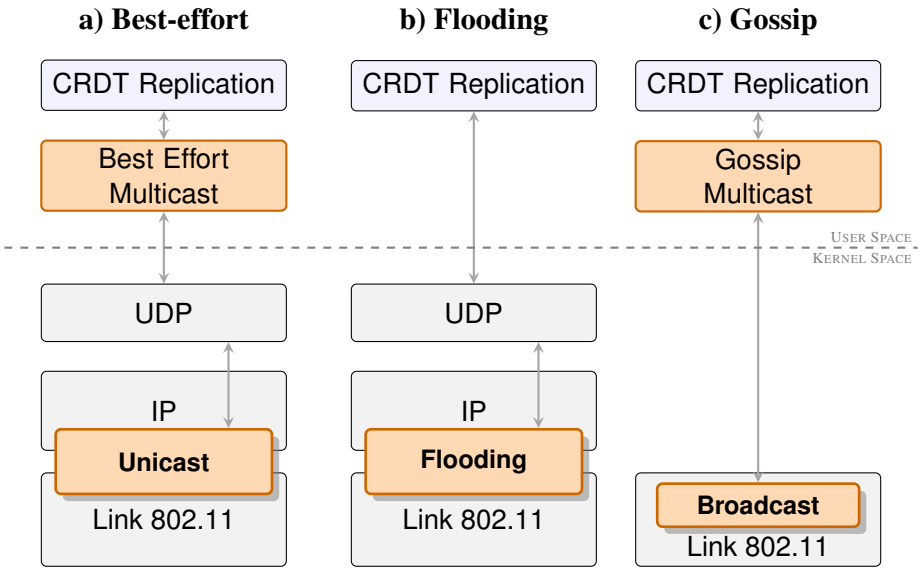


Figure 2. Three co-designed network stacks for efficient replication synchronisation.

Figure 2a depicts the design of the network stack for best-effort multicast. It requires the standard unicast communication and multi-hop routing provided by the IP and link layers suitable for wireless ad hoc networks. In addition to that, we assume that flooding multicast, depicted in Figure 2b, requires a flooding primitive available in the IP and link layers to reach all drones in a swarm in a controlled manner. Finally, Figure 2c shows the guidelines to design a gossip multicast that requires broadcast communication from the link layer only. In Subsection 3.4, we provide implementation details of our CRDT-network co-design.

3.4. Implementation

CRDT-based replication. We implemented our CRDT-based replication scheme using a Grow-only Counter (GCounter). This distributed data structure allows drones to replicate information collected during an exploratory mission. We used delta-based CRDT to ensure a lightweight implementation, as described in Subsection 3.2. For simplicity, each GCounter operation increments a replicated counter locally, which yields a monotonic and continuously interpretable notion of progress. This enables us to define per-node convergence as the fraction of the global count already incorporated by a replica.

Unlike typical reactive applications, our implementation decouples operation execution from replica synchronisation. Regardless of the local operation rate, the implementation aggregates all local increments into a single cumulative delta. This aggregated state is periodically sent to all drones at a configured periodic interval T. This parameter controls the synchronisation frequency, acting as a flow control mechanism that prevents high-frequency application bursts from saturating the network interface.

Network-layer multicast implementations. To implement best-effort and flooding multicast primitives, we leverage `batman` routing protocol as the underlying communication middleware. Leveraging the latest stable version (V), which is integrated into the Linux kernel (`batman-adv`), this approach simplifies deployment and minimises user-space context switching overhead compared to other dissemination solutions. Figure 3 shows `batman` as a cross-layer routing protocol.

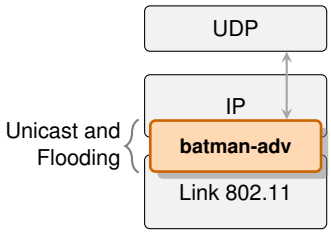


Figure 3. batman-adv routing protocol.

Best-effort multicast implementation. Based on multi-hop routing and unicast communication capabilities of `batman-adv`, we implemented a simple best-effort multicast protocol. We assume that each drone has access to a simple membership system which maintains the list of the drones in the swarm. Whenever a drone sends a synchronisation message at time T, it addresses it to each drone of the list using unicast communication. Since drones only disseminate their own local deltas for synchronisation, this approach is costly as it requires sending N^2 messages for a swarm of N drones.

Flooding multicast implementation. `batman-adv` has a flooding functionality that allows us to use it as communication middleware for multicasting. In this multicast implementation, an application instance of a drone sends synchronisation messages to all drones of the swarm via a flooding IP address. At the Linux kernel level, `batman-adv` controls the dissemination of synchronisation messages and handles the flooding mechanism across the network. This approach dramatically reduces the application-layer overhead at the source compared to the best-effort one. However, this approach also strongly relies on IP multi-hop routing to function properly, which is irrelevant for CRDT replication.

Application-layer multicast implementation. To disseminate synchronisation messages without multi-hop routing, we implemented a purely application-layer multicast approach based on Reliable Probabilistic Information Dissemination, RAPID [Drabkin et al. 2007], a gossip protocol for mobile wireless ad hoc networks. Whenever a drone sends a synchronisation message at the interval T, our implementation simply multicasts the message

to all drones in the communication range leveraging the broadcast communication primitive of the link layer. Then, each recipient drone uses a counter-based algorithm to decide if it acts as a synchronisation message forwarder or not. The counter-based algorithm allows drones to keep track of the number of neighbours that have previously decided to forward it in case it has seen fewer than M such gossips. More precisely, the technique adopts a context-aware dissemination model that combines probabilistic forwarding with deterministic corrective mechanisms. The main implemented procedures are:

1. **Metadata-Driven Recovery:** Drones first disseminate compact metadata (gossip) describing the updates they hold. If a node detects that it is missing a particular update, it explicitly requests the corresponding data from its neighbours, enabling reliable recovery without continuous flooding.
2. **Density-Aware Suppression:** To mitigate broadcast storms, the protocol dynamically adjusts its forwarding probability based on local neighbourhood density. Drones in crowded regions reduce their probability of rebroadcasting messages, while corrective timers ensure eventual retransmission when no neighbour forwards a message. This design allows our implementation to scale more efficiently than naive flooding while preserving high reliability.

4. Evaluation

We evaluate in this section the performance of CRDT replication atop drone swarms. To better understand the challenges in synchronising replicas, we first characterise the communication network. Then, we assess the efficiency and the scalability of the different approaches to disseminate synchronisation messages.

4.1. Methodology

All experiments were conducted using the MACE emulation framework [Ferreira et al. 2021], which enables reproducible experiments with mobile ad-hoc distributed applications by executing real binaries (native processes, containers, or VMs) inside isolated nodes while a network emulator dynamically enforces wireless connectivity based on node positions and communication range. Node mobility is controlled by mobility models or external controllers, and the framework facilitates experiment instrumentation (e.g., packet tracing) during runtime.

Evaluation metrics. Our two primary metrics to quantify performance of different scenarios are:

1. **Synchronisation percentage:** Defined as the average percentage of the global state that has been successfully replicated to each node by the end of the synchronisation phase. Mathematically, let S_{global} be the set of all unique operations generated in the system and S_i be the set of operations held by node i. The convergence ratio is calculated as $\frac{1}{N}\sum_{i=1}^{N}\frac{|S_i|}{|S_{global}|}$. This metric quantifies the reliability of the dissemination strategy in ensuring data consistency.
2. **Network Overhead:** Measured as the total number of packets transmitted and received at the data-link interface. We specifically selected packets over application-level messages to capture the total cross-layer cost, encompassing both the application payload and the underlying routing control traffic (e.g., routing beacons, gossip metadata, and control exchanges) which would otherwise remain invisible

in a message-count metric. This serves as a direct proxy for channel contention and energy expenditure.

Each experiment is run 10 times. To compute the primary metrics, we report the mean and standard deviation across these runs. Error bars refer to 95% confidence intervals computed using the Student-t distribution.

Experimental Setup. The simulated environment consists of a square area adjusted according to the scenario requirements. The wireless medium emulates an IEEE 802.11 ad-hoc network with a nominal bandwidth of 11 Mbps and a communication range of approximately 160 metres.

For generality, we make no assumption about the mobility pattern of the drones. To ensure reproducibility, drones move at slightly varying speeds ranging from 4 to 4.2 m/s according to the Random Waypoint model [Camp et al. 2002]. This speed range is consistent with small to medium-sized drone platforms in cooperative exploratory missions, where moderate and relatively stable motion is typically assumed. For most of the experiments, we consider a fixed, squared exploration area of one km^2. Moreover, the swarm size varies from 10 to 50 drones.

The experimental timeline is divided into two distinct phases:

- **Workload Phase:** During this phase, which lasts 10 seconds, each drone executes locally two operations per second to modify the replicated CRDT and sends one synchronisation message per second to the network.
- **Synchronisation Phase:** This second phase, which lasts 10 additional seconds, allows drones to continue synchronising their states. To this end, they continue sending one synchronisation message every second to the swarm.

4.2. Drone swarms' network characteristics

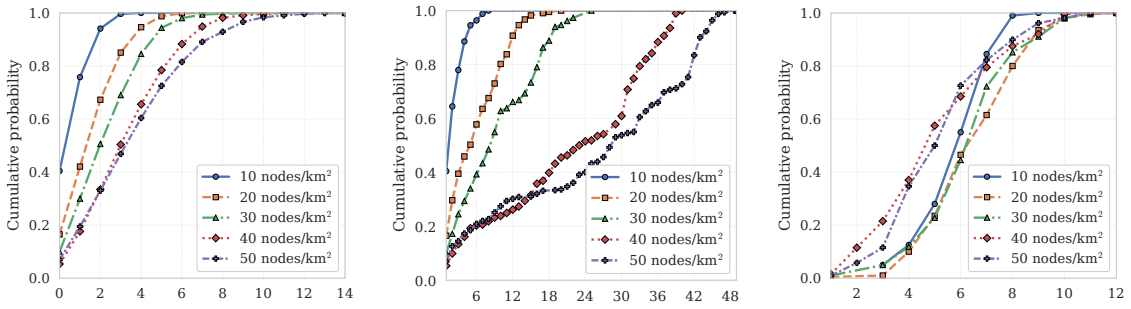
To better understand the challenges in replica synchronisation, we first characterise the communication network connectivity. To do that, we vary the network density from 10 to 50 nodes/km^2. For each experiment, we periodically (every second) collected drone positions in order to compute the neighbourhood degree of a drone, the connected component size and the number of partitions in a swarm. Each experiment is run 10 times.

Neighbourhood degree. Figure 4a shows the empirical cumulative distribution function (ECDF) of the neighbourhood degree of drones for five different network densities. The probability of having no one-hop neighbour remains non-negligible in sparser settings (e.g., $P[\deg = 0] \approx 40.45\%$ at 10 nodes/km^2), but it drops substantially at higher densities (about 5–7% for 40–50 nodes/km^2). The median degree increases from 1 (10 nodes/km^2) to 4 (50 nodes/km^2), and the 90th percentile increases from 2 to 8, indicating a progressively more redundant one-hop neighbourhood.

Connected component size. To estimate the reachability of multicasting a single synchronisation message, we collected the size of connected components. Figure 4b shows the ECDF of these measurements. We observe that higher densities substantially increase the likelihood of belonging to larger components. In particular, the probability of being isolated in a singleton component ($\mathrm{comp} = 1$) decreases from about 40.45% at 10

nodes/km^2 to the single-digit range at higher densities. The median component size grows from 2 (10 nodes/km^2) to values on the order of tens for 30–50 nodes/km^2, and the upper tail increasingly concentrates near large fractions of the swarm (e.g., 90th-percentile component sizes ranging from 19 to 43 nodes depending on the density).

Number of network partitions. To complement reachability of a single message multicast, we measure the total number of connected components, *i.e.*, network partitions. Intuitively, the higher the number of partitions, the worse the communication. Figure 4c shows that fully connected swarms are rare in such sparse networks (typically around $\sim 1\%$ of samples for 20–50 nodes/km^2, and not observed at 10 nodes/km^2 within our sampling window). Across densities, the median number of partitions is typically between five and seven, providing a compact global view that is consistent with the component-size distributions above. This metric helps contextualize why retransmission-capable approaches can sustain higher coverage when links are intermittent: dissemination must traverse a time-varying set of partitions over the workload and synchronisation phases.



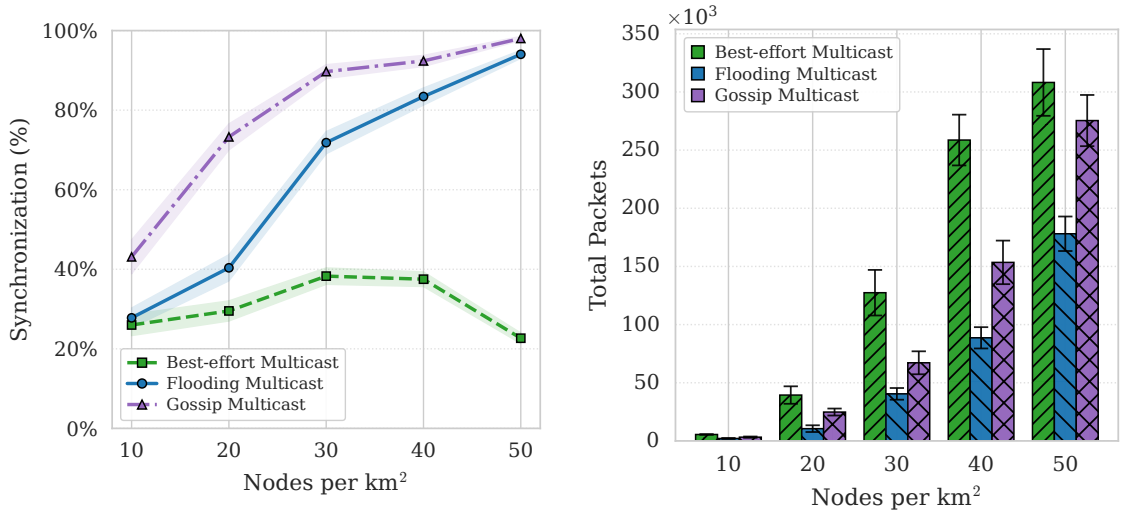
(a) Immediate neighborhood degree. **(b) Connected component size.** **(c) Number of partitions.**

Figure 4. ECDFs on densities from 10 to 50 nodes/km^2.

4.3. Efficiency and network overhead of multicast primitives

Figure 5a shows that the synchronisation percentage increases with density for flooding and gossip multicast. *Gossip multicast* achieves near-full replica synchronisation in the densest setting (98.0% at 50 nodes/km^2), confirming that higher neighbour redundancy yields higher, more efficient replica synchronisation. *Flooding multicast* also strongly benefits from denser settings, rising from 27.8% at 10 nodes/km^2 to 94.0% in 50 nodes/km^2. In contrast, *Best-effort multicast* performs poorly in sparse networks, with synchronisation percentages well below 50%, peaking at 38.3% in 30 nodes/km^2.

The packet-level network overhead reported in Figure 5b shows the cost of replica synchronisation. *Best-effort multicast* exhibits a sharp overhead increase, exceeding 308,000 packets in the highest density. Moreover, *Gossip multicast* incurs a higher network overhead than *Flooding multicast*. For instance, the former transmitted roughly 97,000 more packets than the latter in a density of 50 nodes/km^2. Overall, *Flooding multicast* yields the lowest packet overhead across densities, at the expense of weaker synchronisation efficiency in sparse networks.



(a) Synchronisation (capacity).

(b) Total packets.

Figure 5. Comparing synchronisation efficiency and network overhead.

4.4. Scalability

Figure 6 shows the synchronisation percentage with varying number of nodes and a density fixed to 30 nodes/km^2. We observe that the efficiency of replica synchronisation decreases as the area increases. The efficiency of *Flooding multicast* drops from 88.7% with 10 nodes to 47.5% with 50 nodes. The performance of *Gossip multicast* remains higher across the entire range, reaching 71.7% with 50 nodes. In contrast, *Best-effort multicast* degrades more sharply and reaches 17.5% with 50 nodes.

This result corroborates the findings of dissemination analysis over an intermittently connected topology, as characterized in Subsection 4.2. As the area becomes larger, synchronisation messages are likely to traverse multiple unstable wireless links, which, in turn, increases the number of omission faults. It suggests that the forwarding mechanism of *Gossip multicast* is much more efficient than that of *Flooding multicast*.

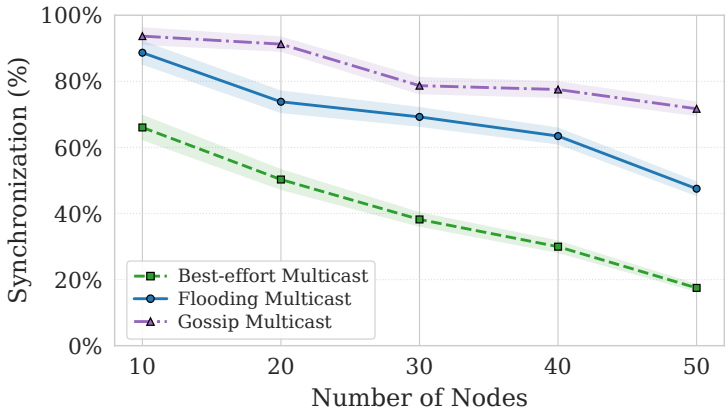


Figure 6. Comparing the scalability of multicast primitives.

5. Discussion and Limitations

In this section, we briefly discuss additional observations and open questions. Then, we highlight the limitations of our approach along with directions for future work.

5.1. Sensitivity to packet loss and synchronisation frequency

To evaluate sensitivity to packet loss, we injected loss rates of 2%, 4%, 6%, 8%, 10%, and 20%. Overall, the qualitative behaviour of the synchronisation results remained essentially unchanged, which suggests that the synchronisation percentage metric is not sensitive to a small to moderate packet loss rate. Unsurprisingly, a bigger impact was observed in the network overhead metric: the total number of packets decreased proportionally to the packet loss rate. These findings are aligned with previous work [Johnson et al. 2008, Drabkin et al. 2007], where dissemination performance losses are mostly due to the mobility of nodes, multi-hop routing and dynamically changing topologies.

To evaluate the sensitivity to synchronisation frequency, we doubled it, from one to two synchronisations per second. Regardless of the dissemination approach, we observed that the performance gains were small and similar. However, a higher synchronisation rate considerably increased the network overhead. For this reason, we chose a relatively low synchronisation rate, based on previous work [Enes et al. 2019], which allowed us to clearly expose the trade-off between replica synchronisation efficiency and network overhead.

5.2. Limitations and future work

Our work compared three promising multicast techniques and a relevant type of CRDT as a replicated data structure. We would like to expand our study to compare a larger number of different multicast primitives (*e.g.*, different flavours of gossip) and also to consider different types of CRDTs.

As mentioned in Section 4.1, we considered only Random Waypoint as a mobility model for its simplicity and to ensure the reproducibility of our experiments. Future work should assess whether our findings also hold for different, more specific mobility models.

Finally, our evaluation focused on common omission faults that happen mostly due to drone mobility in a sparse communication network. Considering a Byzantine failure model and more advanced fault injection techniques are both worthwhile future work.

6. Conclusion

Drone swarms are emerging as promising platforms to build novel applications with great societal impact, ranging from wildlife remote sensing to environmental hazard monitoring. For timely coordination, drones in a swarm need to continuously exchange information. To this end, system designers can leverage data replication, a fundamental building block of real-world distributed computing. To be useful in practice, replication needs to perform efficiently, yet provide a high level of fault tolerance. However, data replication in wireless mobile ad hoc networks, where network partitions are common, may suffer from low performance or can lead to data unavailability. In this work, we introduce a highly available replication scheme based on CRDTs, suitable for drone swarms, that trades consistency for availability. We propose a system architecture and implementation

guidelines that explore different CRDT/network co-designs. Through extensive emulations, we characterised the underlying communication network and evaluated the performance of CRDT replica synchronisation using three different multicast primitives, namely best-effort, flooding and gossip multicast. Our evaluation shows that gossip multicast co-designed with a lightweight network stack provides the best performance and scalability. Nonetheless, flooding multicast remains competitive, especially for small swarms. We hope that our findings on replication in drone swarms will open up new directions for distributed computing research in wireless mobile ad hoc systems.

Acknowledgments

We thank the reviewers for their constructive comments. This work was supported by the CAPES/BRAFITEC project no. 88887.917033/2023 and by CDEFI.

References

Camp, T., Boleng, J., and Davies, V. (2002). A survey of mobility models for ad hoc network research. *Wireless communications and mobile computing*, 2(5):483–502.

Chung, T. H., Clement, M. R., Day, M. A., Jones, K. D., Davis, D., and Jones, M. (2016). Live-fly, large-scale field experimentation for large numbers of fixed-wing uavs. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1255–1262. IEEE.

Drabkin, V., Friedman, R., Kliot, G., and Segal, M. (2007). Rapid: Reliable probabilistic dissemination in wireless ad-hoc networks. In *2007 26th IEEE International Symposium on Reliable Distributed Systems (SRDS 2007)*, pages 13–22. IEEE.

Enes, V., Almeida, P. S., Baquero, C., and Leitão, J. (2019). Efficient synchronization of state-based crdts. *arXiv preprint arXiv:1803.02750*.

Ferreira, B. C., Dufour, G., and Silvestre, G. (2021). Mace: A mobile ad-hoc computing emulation framework. In *2021 International Conference on Computer Communications and Networks (ICCCN)*, pages 1–6. IEEE.

Gilbert, S. and Lynch, N. (2012). Perspectives on the cap theorem. *Computer*, 45(2):30–36.

Gomes, P. S., Rodrigues, M. B., and Baquero, C. (2025). Conflictsync: Bandwidth efficient synchronization of divergent state. *arXiv preprint arXiv:2505.01144*.

Guillen-Perez, A., Montoya, A.-M., Sanchez-Aarnoutse, J.-C., and Cano, M.-D. (2021). A comparative performance evaluation of routing protocols for flying ad-hoc networks in real conditions. *Applied Sciences*, 11(10):4363.

Hattenberger, G., Bronz, M., and Gorraz, M. (2014). Using the Paparazzi UAV System for Scientific Research. In *IMAV 2014 Proceedings*, pages 247–252, Delft, Netherlands.

Herrmann, C., Mager, F., and Zimmerling, M. (2018). Mixer: Efficient many-to-all broadcast in dynamic wireless mesh networks. In *Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems*.

Horyna, J., Baca, T., Walter, V., Albani, D., Hert, D., Ferrante, E., and Saska, M. (2023). Decentralized swarms of unmanned aerial vehicles for search and rescue operations without explicit communication. *Autonomous Robots*, 47(1):77–93.

Hossein Motlagh, N., Zaidan, M. A., Irjala, M., Rebeiro-Hargrave, A., Nurmi, P., and Tarkoma, S. (2025). Drones in the sky: Air quality monitoring at heights. In *Proceedings of the 23rd Annual International Conference on Mobile Systems, Applications and Services*, pages 717–722.

Ivošević, B., Han, Y.-G., Cho, Y., and Kwon, O. (2015). The use of conservation drones in ecology and wildlife research. *Journal of Ecology and Environment*, 38(1):113–118.

Janic, M. (2005). *Multicast in network and application layer*. Dissertation (tu delft), Delft University of Technology.

Johnson, D., Ntlatlapa, N. S., and Aichele, C. (2008). A simple pragmatic approach to mesh routing using BATMAN. In *2nd IFIP International Symposium on Wireless Communications and Information Technology in Developing Countries*.

Levis, P., Brewer, E., Culler, D., Gay, D., Madden, S., Patel, N., Polastre, J., Shenker, S., Szewczyk, R., and Woo, A. (2008). The emergence of a networking primitive in wireless sensor networks. *Communications of the ACM*, 51(7):99–106.

Preiss, J. A., Honig, W., Sukhatme, G. S., and Ayanian, N. (2017). Crazyswarm: A large nano-quadcopter swarm. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3299–3304. IEEE.

Saquib, N., Krintz, C., and Wolski, R. (2021). Log-structured conflict-free replicated data types. Technical Report 2021-01, UCSB Technical Report.

Schranz, M., Di Caro, G. A., Schmickl, T., Elmenreich, W., Arvin, F., Şekercioğlu, A., and Sende, M. (2021). Swarm intelligence and cyber-physical systems: Concepts, challenges and future trends. *Swarm and Evolutionary Computation*, 60:100762.

Shah, D. (2009). Gossip algorithms. *Foundations and Trends in Networking*, 3(1):1–125.

Shapiro, M., Preguiça, N., Baquero, C., and Zawirski, M. (2011). Conflict-free replicated data types. In *Symposium on Self-Stabilizing Systems*, pages 386–400. Springer.

Spica, R., Giordano, P. R., Ryll, M., Bülthoff, H. H., and Franchi, A. (2013). An open-source hardware/software architecture for quadrotor uavs. *IFAC Proceedings Volumes*, 46(30):198–205.

Vásárhelyi, G., Virágh, C., Somorjai, G., Tarcai, N., Szörényi, T., Nepusz, T., and Vicsek, T. (2014). Outdoor flocking and formation flight with autonomous aerial robots. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3866–3873. IEEE.

Villa, T. F., Gonzalez, F., Miljievic, B., Ristovski, Z. D., and Morawska, L. (2016). An overview of small unmanned aerial vehicles for air quality measurements: Present applications and future prospectives. *Sensors*, 16(7):1072.

Weinstein, A., Cho, A., Loianno, G., and Kumar, V. (2018). Visual inertial odometry swarm: An autonomous swarm of vision-based quadrotors. *IEEE Robotics and Automation Letters*, 3(3):1801–1807.

Yuan, C., Zhang, Y., and Liu, Z. (2015). A survey on technologies for automatic forest fire monitoring, detection, and fighting using unmanned aerial vehicles and remote sensing techniques. *Canadian Journal of Forest Research*, 45:783–792.