Mitigating Latency and Partitioning through Size Regulation in Blockchain-Enabled Robot Swarms

Raina Zakir[1 [0000−0002−2419−3789]], Marco Dorigo[1 [0000−0002−3971−0507]], and Volker Strobel[1 [0000−0003−2974−9827]]

IRIDIA, Université Libre de Bruxelles, Brussels, Belgium
{raina.zakir,volker.strobel}@ulb.be, mdorigo@ulb.ac.be

Abstract. Blockchain technology has recently been integrated into robot swarms, providing the benefits of secure decentralized coordination and increased resilience against malicious agents. The security of blockchain technology relies on the consensus protocol, which ensures data consistency across the network. In robot swarms, however, changing network topologies and communication constraints can cause severe network partitioning. When such partitions occur, block production may be delayed or even halted, hindering the dissemination of information required for time-sensitive applications. In this work, we address the issue of delayed block production through adaptive swarm size control. We provide the first proof of concept for open swarms that self-regulate their swarm size during operation, based on acceptable block production delays. Our simulation results demonstrate that adaptive swarm size regulation effectively reduces latency, enabling the swarm to adapt its block production rate to acceptable task execution delays.

1 Introduction

Robot swarms are decentralized systems, typically composed of a large number of autonomous robots that coordinate to perform tasks [5, 6]. Within such systems, self-organized collective behavior emerges from local interactions among the robots themselves and with their surrounding environment. The decentralized nature of robot swarms facilitates parallel task execution, which can provide fault tolerance against individual failures. However, studies have shown that robot redundancy and parallelization are not sufficient to achieve system robustness against misbehaving robots [29, 30].

Recent research indicates that blockchain technology can improve the Byzantine fault tolerance of robot swarms by effectively identifying and neutralizing the actions of Byzantine robots (those whose behavior differs from their intended function). For instance, Strobel et al. [28] demonstrated the potential of blockchain-enabled robot swarms (using large swarms of up to 120 simulated robots) to maintain consensus in the presence of Byzantine agents through a collective sensing problem. Moroncelli et al. [15] showed that a blockchain-secured robot swarm can perform Simultaneous Localization and Mapping (SLAM) in unknown environments while remaining resilient to Byzantine robots. Van Calck

et al. [34] further presented a blockchain-based information market, in which robots trade data and penalize Byzantine behavior in a foraging scenario. Zhao et al. [41] proposed a generic framework that leverages the programmability of smart contracts to achieve consensus within a robot swarm in the presence of Byzantine robots, and illustrated its application in a foraging scenario. In these works, the blockchain is maintained collectively by the robots, with each robot functioning as a blockchain node in the decentralized network with its own version of the blockchain (see Section 2 for fundamentals of blockchain technology).

A blockchain consensus algorithm enables the swarm to reach an agreement on the validity of transactions and to add new blocks to the chain accordingly. Even though several consensus algorithms have been proposed over the years [17, 35, 36], most studies on blockchain-enabled robot swarms have adopted Proof-of-Authority (PoA) due to its lightweight design and computational efficiency [21, 28, 9, 41, 19]. Given the infancy of blockchain-enabled robot swarms, hardly any research has addressed issues of the underlying consensus protocol, in particular, the issue of delays in block production when using PoA. Recently, Simionato et al. [27] presented a study examining the security of blockchain-enabled robot swarms under network partitioning and proposed a framework for improving blockchain data consistency and enhancing the reliability of swarm coordination. This framework incorporates trigger rules (detection of delayed block production) and panic behaviors (temporarily increasing the robots' communication range) to reduce blockchain inconsistencies and unavailability in sparse networks. While temporarily increasing the communication range can be effective, it has limitations in real-world applications. For example, a panic behavior that switches the communication range from local (e.g., using Bluetooth) to more global (e.g., using Wi-Fi) may lead to high energy consumption and requires suitable robot hardware.

As an alternative, we propose introducing heterogeneity into the swarm by adding *relay robots*—lightweight and inexpensive blockchain nodes that enhance communication within the network. These relay nodes observe the environment and generate meaningful data, but do not produce blockchain blocks, as their purpose is to help facilitate data propagation and connectivity between otherwise sparse robots or network partitions. Relay robots can feature relatively larger, but still local communication ranges compared to block producers (called sealers in PoA, see Section 2), ensuring that transactions, messages, and blockchain updates continue to propagate steadily throughout the swarm even when direct links between sealers are unavailable. Depending on the application, relay robots can be integrated into the swarm in various ways, such as ground-based robots, aerial drones, or even stationary modules capable of maintaining robust communication links across the operational area.

In this work, we show that regulating swarm size through the addition or removal of ground-based relay robots enables the tuning of block production time according to application requirements. In general, swarm size regulation enables the adoption of different collective strategies based on the estimated group size, which may depend on factors such as speed, risk, or task complexity [14, 10, 13].

Although several studies have addressed group size regulation [8, 2, 18], in all these works, group size is typically assumed to be known a priori, disregarding problems that can occur during swarm operation, such as malfunctioning and malicious robots.

Swarm-size regulation also supports the scalability principle of swarm robotics by enabling the swarm to autonomously adjust its size based on system performance or task demands. In swarm robotics, scalability refers to the ability of decentralized controllers to operate effectively across different swarm sizes and adapt dynamically to changes caused by failures, malicious robots, or environmental factors. However, current blockchain-enabled swarm robotics implementations lack such adaptive scalability mechanisms.

To address this, we demonstrate adaptive swarm-size regulation and thus provide the first proof of concept of *open swarms* [30, 34] where robots can join or leave the system (as opposed to traditional closed swarms, where the number of robots remains fixed). We implement and analyze the performance of our swarm using the ARGoS robot swarm simulator [24], demonstrating how relay robots can mitigate delays in block production of the robots' blockchains.

2 Blockchain Fundamentals

Blockchain technology enables peer-to-peer networks to reach consensus on shared data without requiring trust among participating nodes [16, 1]. The blockchain is a decentralized ledger composed of sequentially linked blocks, beginning with a common genesis block. Each block records data generated by nodes called *transactions* and includes a cryptographic hash referencing its predecessor. By leveraging blockchain technology, tamper-proof algorithms can be executed in a distributed manner by the use of *smart contracts*. Robot nodes exchange information via blockchain transactions and can establish system-wide rules encoded within these smart contracts, enabling the distributed control and coordination of swarm behavior.

Using the Proof-of-Authority (PoA) consensus protocol [31], a designated set of N nodes, known as *sealers*, create new blocks at fixed intervals. After proposing a block, a sealer must wait for at least $\lfloor \frac{N}{2} \rfloor + 1$ subsequent blocks before producing another block. When multiple sealers generate blocks simultaneously (but potentially in different partitions), conflicting versions of the blockchain may arise due to differences in transactions or their ordering, resulting in what are known as *forks*. To resolve such conflicts, nodes agree on the chain with the highest difficulty: a difficulty of 2 is assigned to blocks produced by the preferred sealer (for each block, a preferred sealer is selected in a round-robin manner) and a difficulty of 1 to those produced by non-preferred sealers. Any attempt to modify an existing block in a blockchain invalidates all subsequent blocks (and, therefore, reduces the chain's overall difficulty), ensuring data integrity and tamper resistance.

Another characteristic of PoA is the minimum time required between consecutive blocks, known as the *block period*, which we set to a default of 15 s,

following previous studies [28, 27, 21]. Setting the block time involves a trade-off: shorter block times enable real-time coordination but increase the risk of forks, causing temporary inconsistencies. Setting a minimum block period of 15 s ensures that the frequency of block production does not become excessively high, which would otherwise lead to greater communication, computation, and data storage costs, as well as to a higher rate of blockchain forks. The time elapsed between the creation of two successive blocks is referred to as the *block production time*, and the difference between the block production time and the block period defines the *block production delay*. In blockchain-enabled robot swarms, the block production time frequently exceeds the block period, as, due to the sparse connectivity and changing network topology of a robot swarm, a suitable block sealer is often temporarily unavailable.

One of the main challenges of integrating blockchain technology into a robot swarm is the constantly changing network topology of a swarm [7]. Blockchain technology was originally designed for networks of stationary nodes; however, in a swarm of robots, the nodes are mobile. Due to their limited communication range, robots can propose and validate new blocks only with nearby peers. Local communication, combined with potential malfunctions or breakdowns inherent to mobile systems, can result in disconnections among robots within the swarm. Such disconnections may disrupt blockchain operations by affecting block production, causing latency or even a complete halt in block generation—especially in PoA systems, where maintaining communication among sealers is essential for consensus. In fact, PoA relies on a small, fixed set of sealers that must exchange messages to propose, validate, and finalize blocks. Weak network links between sealers can prevent these messages from being transmitted efficiently, ultimately causing the chain to stall.

According to the CAP theorem, a distributed system experiencing a network partition (P) can only guarantee either consistency (C) or availability (A) [3]. When network connectivity is sparse, PoA is designed to prioritize consistency over availability, choosing to delay block production. While this delayed block production avoids excessive forks, it introduces latency that can impact task execution. Latency in block production is particularly critical in applications where robots rely on timely blockchain updates to coordinate actions or make decisions, rather than merely using the blockchain for data logging. Most previous approaches have relied on smart contracts to handle Byzantine faults in robot swarms [41, 28]. However, these approaches fail to maintain availability under severe network partitioning—a situation that can also be exploited by an attacker to gain control of a relative majority of robots in the swarm.

An alternative approach to resolving network partitions and latency is to utilize distributed ledger technology frameworks that maintain transaction processing and consensus even during network partitions, such as DAG-based architectures [26, 23] that allow concurrent transaction validation and asynchronous consensus without relying on sequential block production. For instance, the DAG-based distributed ledger framework IOTA includes a smart contract layer that operates on subnetworks, allowing contracts to execute independently while

committing their results to the main chain, supporting both partition tolerance and parallel execution [25, 12]. Similarly, SwarmDAG begins with multiple isolated blockchain views that emerge during network partitions and later merges them into a single DAG ledger once connectivity is restored, thereby preserving all locally collected data [33, 22]. Avalanche, on the other hand, relies on a probabilistic consensus mechanism over a DAG of transactions, where nodes continuously sample and vote among peers until consensus emerges, achieving high throughput and rapid finality without central coordination [32]. Finally, Blockgraph is a DAG-based blockchain architecture specifically designed for mobile ad-hoc networks where consensus, block management, and group management are integrated, enabling nodes to maintain local progress even under intermittent connectivity and to synchronize seamlessly once communication resumes [4, 37]. While DAG-based solutions improve partition tolerance, they face several issues, such as excessive data retention, which is unsuitable for resource-constrained robot swarms, and a lack of the inherent transaction ordering and immutability offered by linear blockchains. Overall, these distributed ledger technology frameworks prioritize partition tolerance over consistency and only offer limited support for smart contracts.

3 Experimental Methods

Scenario In all experiments, we use a blockchain-enabled robot swarm in which the number of sealer robots is fixed during an experiment. The swarm's objective is to perform a task while maintaining a user-defined block production time of x seconds. As a test case, we adopt a collective sensing scenario widely used in previous studies [28, 30, 29, 38, 40, 11]. The swarm aims to reach consensus on the percentage of white tiles in an environment covered with black and white tiles, where white tiles are grouped into two clusters and cover 25 % of the arena. Each robot, equipped with a ground sensor, performs a random walk to estimate the proportion of white tiles. Sealer robots then broadcast their estimates as blockchain transactions within their 10 cm communication range. Relay robots, with an extended communication range of 50 cm, facilitate wider data exchange. A smart contract aggregates all estimate submissions and computes the collective estimate, as implemented in [28].

Setup Our experiments are conducted in the ARGoS robot swarm simulator [24], using Pi-puck robots. The arena is a square environment of 3.61 m^2, surrounded by walls (see Figure 1A and and supplementary video). Unless otherwise stated, each simulation experiment runs for 2,500 timesteps (1 timestep is equivalent to 1 s) and we perform 10 repetitions per setting. Each robot in the swarm functions as a blockchain node and only sealer robots are block producers in Toychain, a lightweight blockchain framework specifically designed for swarm robotics experiments [20]. Toychain implements the Proof-of-Authority consensus protocol, simulating Ethereum's Clique [31]. Its clock synchronization ensures temporal consistency between the ARGoS simulation clock and blockchain operations. By

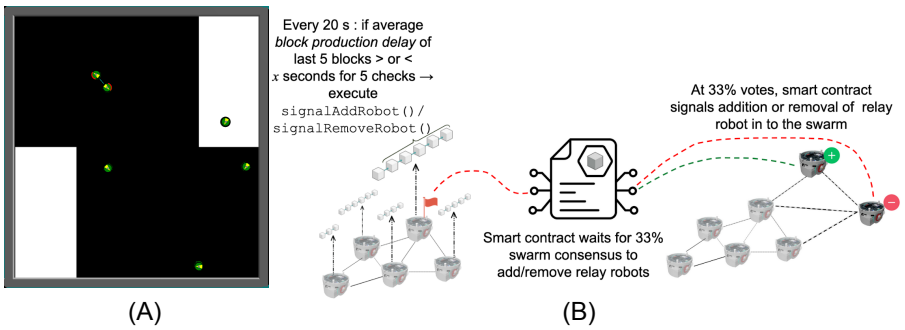

Fig. 1. Experimental setup. (A) Arena setup where Pi-puck robots estimate the proportion of white tiles via random walk. Each robot acts as a blockchain node using Toychain [20] to aggregate estimates. Communication ranges: 10 cm (sealers), 50 cm (relay robots). (B) Schematic of the size-regulation mechanism where robots adjust swarm size via smart contracts to maintain target block times and improve information flow in the collective sensing task in (A).

using Toychain to simulate our blockchain, we obtain results in simulation faster than real-time while ensuring the results are transferable to real-world applications using the mature and established Ethereum framework because of the close modeling of Toychain's PoA after Ethereum's PoA [1]. All the code used for this study is open-source and available as supplementary material [39].

Initialization The robots are randomly distributed in the arena at the start of each experimental run. The robots perform a random walk as in the previous blockchain-enabled swarm robotics studies [28, 30, 29, 27]. To study the regulation of block production time in our system, we conduct three experiments.

In the first experiment, the required block production time x is set to be less than 60 s. We initialize a base swarm of six sealers forming a sparse network and observe how many relay robots are added over time to reduce the block production time to below 60 s.

In the second experiment, we study a swarm of 30 robots (out of which 6 are sealers and 24 are relay robots). We examine how many robots need to be removed from the system to attain a block production time of more than 60 s and whether the swarm converges to a stable size.

In the third experiment, both addition and removal mechanisms are enabled to study how the swarm dynamically maintains the block production time between 50 s and 60 s. In all three experiments, there are six sealers in the swarm that cannot be removed.

As a baseline, we also conduct experiments with fixed swarm sizes of 6 and 24 robots (all of them being sealers) to demonstrate how increasing the swarm size improves block production times and, consequently, task execution times. At the beginning of each run, a new genesis block is created and distributed

to all robots. This block contains the smart contract and the list of authorized sealers participating in the Proof-of-Authority consensus protocol.

Peering In order to synchronize the blocks and transactions, the robot nodes use range-and-bearing sensors to identify peers within the communication range and use the Transmission Control Protocol (TCP) implemented in the Toychain. Using TCP, robots share their node address with other robots within their communication range to establish peer-to-peer connections. Once robots are no longer within communication range, the corresponding connection is terminated.

Robot controller Every 20 s, each robot uses the most recent five blocks of its blockchain version and calculates the average time gap between the blocks to assess the block production time. In the first experiment, if the block production time exceeds 60 s after five consecutive readings, a robot sends a blockchain transaction invoking the `signalAddRobot()` function in the smart contract, signaling that an additional robot should be added to improve performance (see Figure 1B). If 33 % of the swarm flags for addition, a relay robot is introduced in the swarm. Conversely, in the second experiment, if blocks are being produced too quickly—i.e., the block production delay is less than 60 s– a robot invokes the `signalRemoveRobot()` transaction in the smart contract to remove a robot and avoid unnecessary resource usage. If 33 % of the swarm flags for removal, the most recently added relay robot is removed from the swarm. In the third experiment, the robot swarm has to maintain the block production time between 50 and 60 s by adding or removing relay robots.

Performance Metrics We evaluate the performance of our swarm size regulation behavior using the following metrics: (i) the *average block production time* is calculated by recording the rolling average of the last five blocks for each run, and then averaging these rolling averages across all runs of an experiment (ii) the *average number of robots* in the swarm over time (both sealers and relay robots) for all the runs of an experiment; (iii) the average *swarm estimate* computed as the mean of the average estimates from all runs at each time step, representing the collective sensing performance of the swarm; and (iv) the variance and standard deviation of swarm estimates across the runs in each experiment to assess how block production impacts the collective sensing of the environment.

4 Results

Before presenting the performance of our proposed swarm size modulation approach, which uses relay robots to mitigate block production delays, we first analyze how sparse connectivity affects swarm behavior. Figure 2A compares a swarm of 6 sealer robots (in green) and a swarm of 24 sealer robots (in blue) as they evolve over time toward the target mean value of the environment. The plots show both the average swarm estimate and its variability across experimental runs. A key observation is that the larger swarm converges and stabilizes

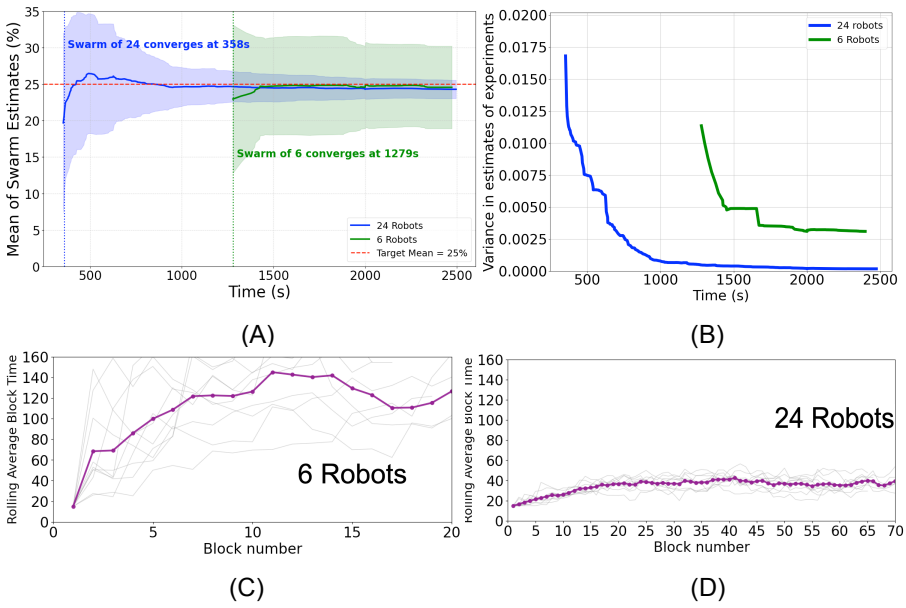

Fig. 2. Baseline experiments. (A) Average swarm estimates (indicating overall sensing performance) and standard deviation (showing variability across runs) over time for two experiments: a sparse swarm of 6 robots and a larger swarm of 24 robots. Vertical lines mark convergence times where the swarm's average remains within ± 5 percentage points of the 25 % actual value. The average swarm estimate is computed from the experimental set at the time when all robots in each run have converged to a mean value. (B) The variance between the runs for 6 robots and a larger swarm of 24 robots. (C–D) Rolling average of block production times for the 6-robot swarm (C) and the 24-robot swarm (D). Each gray line represents one run, while the purple line shows the mean across all runs.

around the target mean of 25% (within ± 5 percentage points) much earlier (at approximately 358 s) compared to the smaller swarm, which stabilizes at approximately 1,279 s. This delay can be attributed to the block production time (evident in Figures 2C and 2D), which slows the propagation of estimates in the sparser swarm. Even after convergence, the smaller swarm exhibits notably higher variability, as shown in the variance plot inset in Figure 2B.

To counter the high block production times observed in sparser swarms (Figures 2C and 2D), we implement the robot controller presented in Section 2 (illustrated in Figure 1B), which monitors block production time and dynamically adjusts swarm size through the addition of relay robots. Assuming an application requirement of maintaining block production time below 60 s, Figure 3A shows how the production time gradually decreases and stabilizes below this threshold as relay robots are introduced in a swarm of six sealers over time. As shown in Figure 3B, on average, 23 robots, including 17 relay robots, are required to

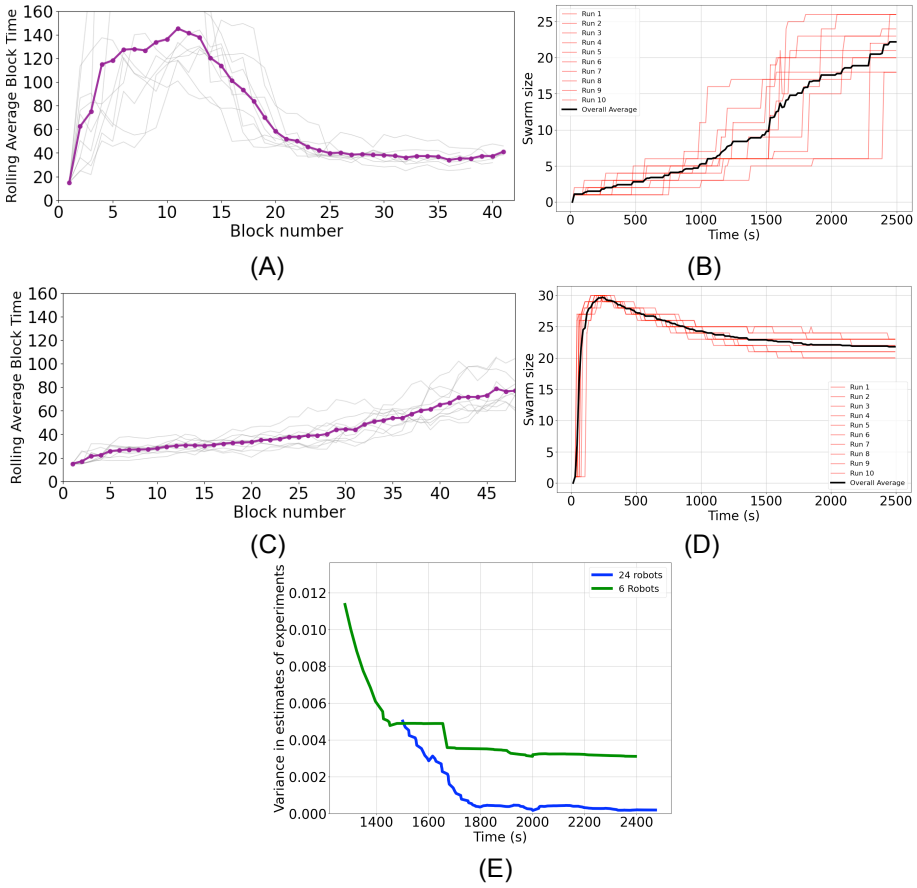

Fig. 3. Experiments 1 and 2. (A, C) Rolling average of block production times of last five blocks (y-axis) as the blocks are produced (x-axis) across multiple runs for two experimental conditions: swarms regulated to keep block time below 60 s (A) and above 60 s (C). In (A), each run begins with six sealer robots, while in (C) the initial swarm includes 30 robots (6 sealers and 24 relay robots). Gray lines represent individual runs, and the purple line shows the average of their rolling averages. (B, D) Evolution of swarm size (y-axis) over time (x-axis) for each run (red) and the averaged swarm size (black) corresponding to the experiments in (A) and (C) obtained from the smart contract. (E) Comparison of variance in swarm estimate (y-axis) over time (x-axis) across individual runs between two cases: a fixed swarm of 6 sealers (experiments in Fig. 2C) and a dynamic swarm where robots can join to maintain block production below 60 s (as in A).

reduce the block production time to below 60 s. Notably, the stabilized block production time is not exactly 60 s but approximately 40 s (Figure 3A). This occurs because robots compute a rolling average of the last five consecutive blocks every 20 s; when older blocks exhibit longer delays, the system tends to add more

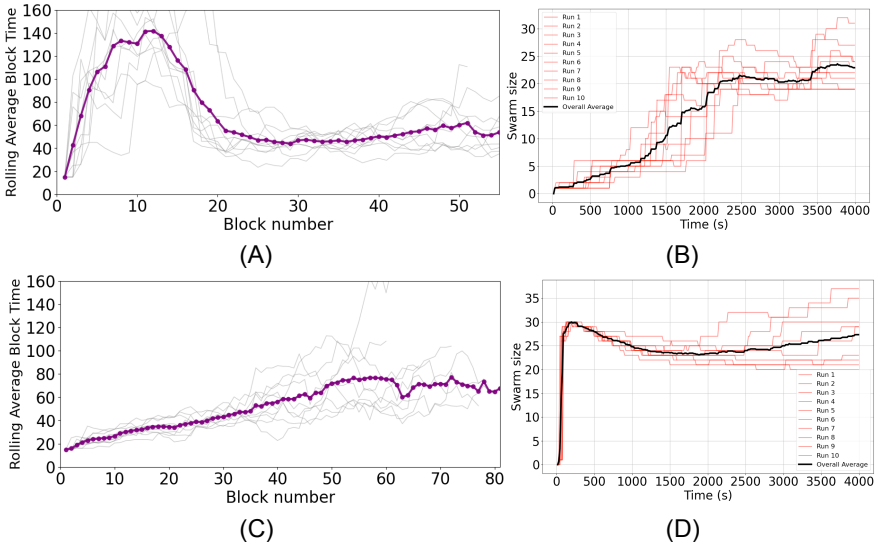

Fig. 4. Experiment 3. (A, C) Rolling average block production times of the last five blocks (y-axis) as the blocks are produced (x-axis) across multiple runs for two experimental setups where swarm size is regulated to maintain block times within 50–60 s. In (A), each run starts with six sealer robots, while in C, the initial swarm includes 30 robots (6 sealers and 24 relay robots). Gray lines show individual runs, and the purple line indicates the average of the rolling averages across runs. (B, D) Evolution of swarm size (y-axis) over time (x-axis) for each run (red) and the average swarm size (black) corresponding to the experiments in (A) and (C) obtained from the smart contract. The experimental runs to plot the figures A-D are run for 4,000 timesteps.

relay robots than strictly necessary. Additionally, sparser swarms tend to trigger the addition of more robots than required, as the transactions that signal robot additions are delayed due to block production delays.

A complementary scenario is shown in Figures 3C and 3D, where the goal is to increase the block production time beyond 60 s by removing relay robots. Also in this case, the final average delay exceeds the target (approximately 75 s) due to the same averaging effect from previous blocks. The average swarm size also converges to approximately 23 robots (7 relay robots are removed). Figure 3E compares the variance across experimental runs for a fixed swarm of six sealers (green; see Figure 2B) and an open swarm that begins with six sealers but gradually adds relay robots (blue). The results clearly show that the swarm incorporating relay robots to mitigate block production delays exhibits lower variance—similar to the larger swarm of 24 robots shown in Figure 2A.

For scenarios requiring precise control of block production delays, consistency must account for delays introduced by block generation. For example, Simionato et al. [27] used the length of blockchain forks as a measure of consistency. If

delays exceed a threshold or exhibit high variance, they can cause undesirable slowdowns in task execution. To address this issue, we integrate dynamic swarm size monitoring, which enables the addition or removal of relay robots to maintain delays within a target range. In Figures 4A and 4C, we set the required block production time between 50 s and 60 s. When the block production time exceeds 60 s, the robots trigger the addition of relay robots; if it falls below 50 s, relay robots are removed. We evaluate this approach in two scenarios: one with a swarm of six sealers and another with a swarm of 30 robots, comprising six sealers and 24 relay robots. In both cases, the block production time stabilizes at approximately 60 s. Figures 4B and 4D (in red) show the swarm size during individual runs, illustrating how it stabilizes while fluctuating in discrete steps to maintain block production within the target range.

5 Discussion and Conclusion

In applications where robots need to change their behavior based on blockchain data—such as voting, coordination, or task execution through smart contracts—timely availability of information across the swarm becomes critical. In this work, we have shown that the availability of such information can not be guaranteed in sparse networks (resulting from low swarm density or large environments), thus leading to task execution delays.

To address this challenge in blockchain-enabled swarms, we introduced the concept of lightweight relay robots that do not seal blocks but instead enhance network connectivity among sealers while generating meaningful task-related data. Such a heterogeneous swarm, composed of both sealers and relay robots, established the connectivity required for reliable information propagation.

We demonstrated that the addition and removal of robots enables a swarm to balance the trade-off between information propagation speed and cost (in terms of number of robots) in a decentralized manner. Our results, therefore, highlight the ability of blockchain-enabled robot swarms to self-regulate their swarm size, enabling them to reach a target block production delay and data dissemination speed. Through this implementation of swarm-size regulation, we also provided a proof of concept for open swarms.

In future work, we aim to develop a comprehensive framework that accounts for different communication ranges for relay robots, supports integration across physically heterogeneous robots, and provides a mechanism to prevent Byzantine robots from joining the swarm when new members are added.

Acknowledgements. RZ, VS, and MD acknowledge support from the Belgian F.R.S.-FNRS, of which they are a FRIA doctoral researcher, a postdoctoral researcher, and a research director, respectively.

Disclosure of interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Buterin, V.: A next-generation smart contract and decentralized application platform. Ethereum project white paper. (2014), https://ethereum.org/en/whitepaper/
2. Cambier, N., Frémont, V., Ferrante, E.: Group-size regulation in self-organised aggregation through the naming game. In: International Symposium on Swarm Behavior and Bio-Inspired Robotics (SWARM 2017) (2017)
3. Carrara, G., Burle, L., Medeiros, D., Albuquerque, C., Menezes, D.: Consistency, availability, and partition tolerance in blockchain: a survey on the consensus mechanism over peer-to-peer networking. Annals of Telecommunications **75**, 163–174 (2020)
4. Cordova, D., Laube, A., Nguyen, T.M.T., Pujolle, G.: Blockgraph: A blockchain for mobile ad hoc networks. In: 4th Cyber Security in Networking Conference (CSNet 2020). pp. 1–8. IEEE (2020)
5. Dorigo, M., Birattari, M., Brambilla, M.: Swarm robotics. Scholarpedia **9**(1), 1463 (2014)
6. Dorigo, M., Theraulaz, G., Trianni, V.: Swarm robotics: Past, present and future. Proceedings of the IEEE **109**(7), 1152–1165 (2021)
7. Dorigo, M., Pacheco, A., Reina, A., Strobel, V.: Blockchain technology for mobile multi-robot systems. Nature Reviews Electrical Engineering **1**(4), 264–274 (2024)
8. Firat, Z., Ferrante, E., Zakir, R., Prasetyo, J., Tuci, E.: Group-size regulation in self-organized aggregation in robot swarms. In: Swarm Intelligence – Proceedings of ANTS 2020 – 12th International Conference. LNCS, vol. 12421, pp. 315–323. Springer (2020)
9. Gupta, H., Strobel, V., Pacheco, A., Ferrante, E., Natalizio, E., Dorigo, M.: Group-level behavioral switch in a robot swarm using blockchain. In: Swarm Intelligence – Proceedings of ANTS 2024 – 14th International Conference. LNCS, vol. 14987, pp. 98–111. Springer (2024)
10. Hamann, H., Reina, A.: Scalability in computing and robotics. IEEE Transactions on Computers **71**(6), 1453–1465 (2021)
11. Kaiser, T.K., Potten, T., Hamann, H.: Evolution of collective decision-making mechanisms for collective perception. In: IEEE Congress on Evolutionary Computation (CEC 2023). pp. 1–8. IEEE (2023)
12. Keramat, F., Peña Queralta, J., Westerlund, T.: Partition-tolerant and Byzantine-tolerant decision making for distributed robotic systems with IOTA and ROS2. IEEE Internet of Things Journal **10**(14), 12985–12998 (2023)
13. Kuckling, J., Luckey, R., Avrutin, V., Vardy, A., Reina, A., Hamann, H.: Do we run large-scale multi-robot systems on the edge? More evidence for two-phase performance in system size scaling. In: IEEE International Conference on Robotics and Automation (ICRA 2024). pp. 4562–4568. IEEE (2024)
14. Mondada, F., Bonani, M., Guignard, A., Magnenat, S., Studer, C., Floreano, D.: Superlinear physical performances in a SWARM-BOT. In: European Conference on Artificial Life. pp. 282–291. Springer (2005)
15. Moroncelli, A., Pacheco, A., Strobel, V., Lajoie, P.Y., Dorigo, M., Reina, A.: Byzantine fault detection in swarm-SLAM using blockchain and geometric constraints. In: Swarm Intelligence – Proceedings of ANTS 2024 – 14th International Conference. LNCS, vol. 14987, pp. 42–56. Springer (2024)
16. Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system (2008), https://bitcoin.org/bitcoin.pdf, Accessed November 9, 2023

17. Nguyen, G.T., Kim, K.: A survey about consensus algorithms used in blockchain. Journal of Information Processing Systems **14**(1), 101–128 (2018)
18. O'Grady, R., Pinciroli, C., Christensen, A.L., Dorigo, M.: Supervised group size regulation in a heterogeneous robotic swarm. In: 9th Conference on Mobile Robots and Competitions (ROBOTICA 2009). pp. 113–119 (2009)
19. Pacheco, A., Strobel, V., Reina, A., Dorigo, M.: Real-time coordination of a foraging robot swarm using blockchain smart contracts. In: Swarm Intelligence – Proceedings of ANTS 2022 – 13th International Conference. LNCS, vol. 13491, pp. 196–208. Springer (2022)
20. Pacheco, A., Denis, U., Zakir, R., Strobel, V., Reina, A., Dorigo, M.: Toychain: A simple blockchain for research in swarm robotics (2024), https://arxiv.org/abs/2407.06630
21. Pacheco, A., Strobel, V., Dorigo, M.: A blockchain-controlled physical robot swarm communicating via an ad-hoc network. In: Swarm Intelligence – Proceedings of ANTS 2020 – 12th International Conference. LNCS, vol. 12421, pp. 3–15. Springer (2020)
22. Peña Queralta, J., Keramat, F., Salimi, S., Fu, L., Yu, X., Westerlund, T.: Blockchain and emerging distributed ledger technologies for decentralized multi-robot systems. Current Robotics Reports **4**, 43–54 (2023)
23. Pervez, H., Muneeb, M., Irfan, M.U., Haq, I.U.: A comparative analysis of dag-based blockchain architectures. In: 12th International Conference on Open Source Systems and Technologies (ICOSST 2018). pp. 27–34. IEEE (2018)
24. Pinciroli, C., Trianni, V., O'Grady, R., Pini, G., Brutschy, A., Brambilla, M., Mathews, N., Ferrante, E., Di Caro, G., Ducatelle, F., Birattari, M., Gambardella, L.M., Dorigo, M.: ARGoS: a modular, parallel, multi-engine simulator for multi-robot systems. Swarm Intelligence **6**(4), 271–295 (2012)
25. Salimpour, S., Keramat, F., Queralta, J.P.n., Westerlund, T.: Decentralized vision-based Byzantine agent detection in multi-robot systems with IOTA smart contracts. In: Foundations and Practice of Security: 15th International Symposium, FPS 2022, Revised Selected Papers. pp. 322–337. Springer (2023)
26. Santos De Campos, M.G., Chanel, C.P., Chauffaut, C., Lacan, J.: Towards a blockchain-based multi-UAV surveillance system. Frontiers in Robotics and AI **8**, 557692 (2021)
27. Simionato, G., Strobel, V., Cimino, M., Dorigo, M.: Analysis and mitigation of inconsistencies in blockchain-enabled robot swarms. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2025), in press. IEEE Press (2025)
28. Strobel, V., Pacheco, A., Dorigo, M.: Robot swarms neutralize harmful Byzantine robots using a blockchain-based token economy. Science Robotics **8**(79), eabm4636 (2023)
29. Strobel, V., Castelló Ferrer, E., Dorigo, M.: Managing Byzantine robots via blockchain technology in a swarm robotics collective decision making scenario. In: Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS 2018). pp. 541–549 (2018)
30. Strobel, V., Castelló Ferrer, E., Dorigo, M.: Blockchain technology secures robot swarms: A comparison of consensus protocols and their resilience to Byzantine robots. Frontiers in Robotics and AI **7**, 54 (2020)
31. Szilágyi, P.: EIP 225: Clique proof-of-authority consensus protocol (2017), https://github.com/ethereum/EIPs/issues/225

32. Team Rocket, Yin, M., Sekniqi, K., Van Renesse, R., Sirer, E.: Scalable and probabilistic leaderless BFT consensus through metastability (2019), https://arxiv.org/abs/1906.08936
33. Tran, J.A., Ramachandran, G.S., Shah, P.M., Danilov, C.B., Santiago, R.A., Krishnamachari, B.: SwarmDAG: A partition tolerant distributed ledger protocol for swarm robotics. Ledger **4** (2019)
34. Van Calck, L., Pacheco, A., Strobel, V., Dorigo, M., Reina, A.: A blockchain-based information market to incentivise cooperation in swarms of self-interested robots. Scientific Reports **13**, 20417 (2023)
35. Xiao, Y., Zhang, N., Lou, W., Hou, Y.T.: A survey of distributed consensus protocols for blockchain networks. IEEE Communications Surveys & Tutorials **22**(2), 1432–1465 (2020)
36. Xu, J., Wang, C., Jia, X.: A survey of blockchain consensus protocols. ACM Computing Surveys **55**, 1–35 (2023)
37. Yang, C., Li, X., Yu, Y., Wang, Z.: Basing diversified services of complex IIoT applications on scalable block graph platform. IEEE Access **7**, 22966–22975 (2019)
38. Zakir, R., Dorigo, M., Reina, A.: Robot swarms break decision deadlocks in collective perception through cross-inhibition. In: Swarm Intelligence – Proceedings of ANTS 2022 – 13th International Conference. LNCS, vol. 13491, pp. 209–221. Springer (2022)
39. Zakir, R.: Supplementary material for paper 'Mitigating Latency and Partitioning through Size Regulation in Blockchain-Enabled Robot Swarms' (2025), Github Repository: https://github.com/rainazakir/size_regulation
40. Zakir, R., Dorigo, M., Reina, A.: Miscommunication between robots can improve group accuracy in best-of-n decision-making. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2024). pp. 9014–9021. IEEE Press (2024)
41. Zhao, H., Pacheco, A., Strobel, V., Reina, A., Liu, X., Dudek, G., Dorigo, M.: A generic framework for Byzantine-tolerant consensus achievement in robot swarms. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2023). pp. 8839–8846. IEEE Press (2023)