Group-Level Behavioral Switch in a Robot Swarm Using Blockchain

Himank Gupta[1,2(✉)], Volker Strobel[1], Alexandre Pacheco[1], Eliseo Ferrante[3], Enrico Natalizio[2,4], and Marco Dorigo[1]

[1] IRIDIA, Université Libre de Bruxelles, Brussels, Belgium
{himank.gupta,volker.strobel,alexandre.pacheco}@ulb.be, mdorigo@ulb.ac.be
[2] Technology Innovation Institute, Abu Dhabi, UAE
enrico.natalizio@tii.ae
[3] Vrije Universiteit Amsterdam, Amsterdam, The Netherlands
e.ferrante@vu.nl
[4] CNRS, LORIA, Université de Lorraine, Villers-lés-Nancy, France

Abstract. In this paper, we introduce the concept of group-level behavioral switch (GLBS) in a robot swarm. We consider two distinct types of GLBS that differ in whether or not the individual robots in the group need to switch their behavior at the same time: the Synchronous GLBS (S-GLBS) and the Asynchronous GLBS (A-GLBS). To implement these GLBSs, we propose a blockchain-based solution built on the Ethereum platform. We then study its performance in terms of required time and success rate in a series of simulation experiments.

1 Introduction

Swarm robotics is the discipline that studies how a large number of robots with relatively limited capabilities plan and coordinate so as to complete complex tasks that are difficult or impossible for individual robots [4,7,9,10]. Swarm robotics is considered as one of the most promising research directions in robotics [37] and it is envisioned that in the future robot swarms will be used in a wide range of possible applications, including environmental monitoring, search and rescue, precision farming, surveillance missions, and extinguishing fires.

Studies in the field have mainly focused on how to implement individual collective behaviors in robot swarms such as aggregation [1], pattern formation [2], self-assembly [14,15,21,27], collective exploration [11,20,39], coordinated motion [13,32], and collective transport [16]. However, for complex real-world applications it is also necessary for the robot swarm to be able to transition from one of the aforementioned behaviors to the next. For instance, during a search and rescue mission, a robot swarm is required to transition between two distinct collective behaviors: first it performs collective exploration to locate casualties and search for survivors, and then it transitions to coordinated transport to bring survivors to a safe place. In this paper, we study how to implement these types of transitions, that we call "group-level behavioral switches", or GLBSs for short, using blockchain technology. Most of the literature in swarm robotics, with the notable exceptions of [6,25,28], has studied collective behaviors in isolation and a versatile way to switch from one collective behavior to

H. Hamann et al. (Eds.): ANTS 2024, LNCS 14987, pp. 98–111, 2024.
https://doi.org/10.1007/978-3-031-70932-6_8

another is missing. In GLBS a certain number, or percentage, of the robots in the swarm have to switch their behavior. Note that, if all the robots in the swarm need to perform the switch then we talk of a swarm-level behavioral switch, or SLBS for short.

A straightforward approach to GLBS would be to develop a centralized controller that monitors the whole swarm and instructs a group of robots to switch behavior when certain conditions are met. However, a centralized controller is a single point of failure for the swarm and is not scalable to larger swarm sizes.

Another way to implement a GLBS involves using a distributed mechanism, such as the one presented in [25] that utilizes a hash-table for disseminating information within the swarm. However, this solution has two drawbacks. First, every individual robot in the swarm can input information into the hash-table at any time, and propagate it to other robots. This can potentially cause a high number of conflicts in information, especially for larger swarm sizes. Second, in their original formulation, there is no explicit mechanism to handle the presence of malicious information, whereas blockchain-based approaches have already been shown to be effective against this issue [31].

In this paper, our objective is to examine the use of blockchain technology to achieve GLBSs in robot swarms. In fact, recent advances in swarm robotics have demonstrated that blockchain technology can be leveraged to achieve fast and conflict-free consensus, even in the presence of Byzantine (faulty or non-cooperating) robots [29,31], and to give instructions to individual robots in the swarm [23]. However, it was not yet studied how to instruct a group of robots to change its behavior in a coordinated way.

The main contributions of this paper are the following:

1. We introduce the group-level behavioral switch (GLBS) notion and classify it into two categories, i.e., asynchronous behavioral switch (A-GLBS) and synchronous behavioral switch (S-GLBS), which differ in whether or not the individual robots need to switch their behavior at the same time.
2. We develop a blockchain-based solution, based on the Ethereum platform, that showcases how the blockchain mitigates the issues of centralized and distributed approaches in the decision-making process for GLBS.
3. We evaluate our blockchain-based solution by conducting experiments on an example scenario chosen so that it includes both A-GLBS and S-GLBS: a simulated fire extinguishing application.[1]

The subsequent sections of the paper are structured in the following manner: Sect. 2 provides a brief review of relevant literature related to decision-making in swarm robotics and the utilization of blockchain technology. Section 3 provides a concise introduction to blockchain technology, smart contracts, and consensus algorithms. In Sect. 4, we present the experimental scenario and the experiment timeline, and provide some details on the experimental setup. Section 5 defines the performance metrics that we use to evaluate the effectiveness of our solution

[1] Note that this is mainly a representative scenario that well illustrates situations in which GLBSs might be needed.

and describes and discusses the results obtained. Finally, conclusions and future works are highlighted in Sect. 6.

2 Related Work

The work presented in this paper is closely related to research on collective decision-making and on swarm-level finite state machines. In fact, the GLBS requires a group of robots in the swarm to make a collective decision to transition to a new state (e.g. of a finite state machine), in response to some event.

Collective decision-making, which is the first step in implementing a GLBS, has been thoroughly studied in the classic, non-blockchain-based robot swarm literature [3,12,33–35]. A limitation of these works when used as the first step for GLBS is that, after convergence, the robots in the swarm do not have explicit knowledge about the swarm status, which is fundamental for implementing a GLBS. Recent advances in the use of blockchain technology to reach consensus in a robot swarm for collective decision-making [8,17,23,31] provide a possible solution to this problem. Indeed, the blockchain, which is maintained in a distributed way by the robots in the swarm, provides a distributed database that allows each of the robots in the swarm to know about the convergence status of any other robot in the swarm. This knowledge can be used to implement our GLBS. However, GLBS was not explicitly studied in that work.

Our implementation is built on the concept of a swarm-level or group-level finite state machine in a robot swarm. We could find only two studies focusing on this concept. The first one introduced the Buzz programming language [25,28], which uses the concept of virtual stigmergy to facilitate coordination within the swarm. This concept is exploited as an approach for executing transitions between different behaviors at the swarm level using a finite state machine (FSM). However, as explained in the Introduction, this approach does not handle well conflicting information nor does it account for the presence of Byzantine robots. The second one proposes a distributed controller also based on a FSM for the allocation of sub-swarms to predefined tasks through a bidding process [6]. However, the study assumes that the swarm is always connected, which makes it unsuitable when this cannot be guaranteed because of low swarm density.

3 Blockchain Fundamentals

A blockchain is a decentralized and distributed ledger that securely records transactions in a transparent and tamper-resistant manner. The primary objective of the original blockchain introduced by Nakamoto [19] was to establish a digital currency system that enables peer-to-peer transactions (i.e., transfers of crypto assets) without the need for external authorities. A blockchain is composed of a sequence of blocks, each of which is a data structure consisting of at least two elements: i) the hash value of the previous block, used to establish the connection and order of the blocks within the blockchain, and ii) a list of transactions.

When a node wants to add a transaction to the blockchain, it sends it to a pool of unconfirmed transactions. To generate a new block, nodes gather a subset of the transactions in the pool, verify each transaction's validity, and bundle the valid transactions in a new candidate block. A consensus algorithm is responsible for deciding if and in which order the new candidate blocks should be added to the blockchain. (Note that the consensus algorithm is always the cause of a delay between the moment a node sends a transaction and the moment in which this transaction is incorporated in the blockchain and disseminated in the network.) Recent research [22,31,38] has demonstrated the feasibility of executing blockchain technology in (real) robot swarms using a lightweight consensus algorithm called proof-of-authority (PoA[2]) [26], which is therefore the consensus algorithm that we use in this work.

In a PoA-based blockchain, the first block, also known as the genesis block, incorporates parameters of the consensus protocol, such as the block interval period, the initial crypto-asset allocation per node, and a list of the permitted sealers (i.e., nodes that are allowed to generate new blocks). With PoA, before being able to propose a new block, sealers need to wait for a certain amount of time (specified by the block interval period) from the time the last block was added to the blockchain. In addition, in a blockchain consisting of N sealers, after a sealer has proposed a new block, it needs to wait for at least $(N/2+1)$ blocks before being able to propose another block. Therefore, at any time, there are no more than $(N-(N/2+1))$ nodes permitted to propose a new block. Forks may occur in the blockchain when more nodes propose a block at the same time. When this happens, the GHOST protocol [36] is used to resolve the forks.

Alongside the utilization of the PoA algorithm, we also make use of smart contracts for the decision making in GLBS. A smart contract is a software program stored in the blockchain and executed by all the nodes in the blockchain network [5]. This mechanism enables decentralized networks to agree on the code, input, and output of software programs. In the context of swarm robotics, smart contracts can act as distributed controllers that allow robots to execute actions based on a consensus in the blockchain network.

4 Methods

4.1 The Experimental Scenario

To evaluate the performance of our blockchain-based approach to the GLBS, we consider a "fire extinguishing" scenario which is representative of any scenarios that require a GLBS (Fig. 1). In this scenario, the robots in the swarm perform a sequence of three behaviors: (i) `detect fires`, (ii) `navigate toward fires`, and (iii) `extinguish fires`. During the first behavior, i.e., fires detection, a robot swarm patrols the environment in search for fires. Upon detection of a fire, in the second behavior, a subset of the robot swarm navigates towards it. Once this group of robots reaches the vicinity of the fire, the robots in the group

[2] https://github.com/ethereum/EIPs/issues/225.

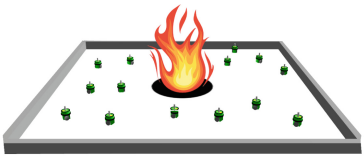

Fig. 1. Our simulation scenario implemented using the ARGoS simulator [24].

switch to the third behavior and start extinguishing the fire. The third behavior in particular has a strong *group-level requirement*, meaning that fire can only be extinguished through the collective action of a group of robots.

In order to complete the sequence of behaviors, the group of robots has to perform two GLBSs: first an asynchronous GLBS (A-GLBS) and then a synchronous GLBS (S-GLBS). The A-GLBS is the switch from `detect fire` to `navigate toward fire`, which starts after the fire has been detected. The S-GLBS is the switch from `navigate toward fire` to `extinguish fire`, which is activated when the required number of robots have reached the fire location.

In the A-GLBS (i.e., from `detect fire` to `navigate toward fire`), the involved robots are not required to switch their behavior simultaneously, as they do not need to coordinate their movement toward the fire. In the S-GLBS (i.e., from `navigate toward fire` to `extinguish fire`), a minimum number of robots need to start shooting water simultaneously for their fire-extinguishing action to be effective. This number, that we call R_{est}, is a parameter of the problem.

In this paper, we consider a specific instance of the generic problem described above; in particular, we assume that: (i) only one fire is present in the experimental arena, (ii) during the fire detection behavior the robots perform a random walk, (iii) the robots share a common frame of reference and a shared clock, and (iv) the minimum cardinality of the set R_{est} of robots necessary to extinguish the fire is $|R_{est}| = 4$. In a more general scenario one could relax some of these assumptions, for example the number of fires and the number of robots necessary to extinguish them could be variable. However, in this paper, our main goal is to show the suitability of our blockchain-based decision making for GLBS: we leave therefore the analysis of more general scenarios for future research.

Even though the swarm could, at least in principle, extinguish the fire using just $|R_{est}|$ robots, a larger swarm has a number of benefits: it reduces the time required to explore the environment to find the fire location, and allows to increase the probability that enough robots are available to execute the GLBSs. In practice, we set the swarm size to $N > |R_{est}|$, and we require that the number of robots involved in each GLBSs is greater than $|R_{est}|$. In particular:

- Let R_{sync} be the set of robots involved in the S-GLBS. We require at least $|R_{sync}| = |R_{est}| + \Delta R_{sync}$ robots, where $|R_{est}|$ is the number of robots necessary for extinguishing the fire and ΔR_{sync} is the number of additional robots necessary to increase the probability of a successful S-GLBS execution.

– Let R_{async} be the set of robots involved in the A-GLBS. We require at least $|R_{async}| = |R_{sync}| + \Delta R_{async}$ robots, where $|R_{sync}|$ is the number of robots necessary for implementing the S-GLBS and ΔR_{async} is the number of additional robots necessary to increase the probability of a successful execution of the A-GLBS.

Therefore, in our solution:

– N robots perform the `detect fire` behavior,
– $|R_{async}|$ robots are involved in the A-GLBS and in the `navigate toward fire` behavior, and
– $|R_{sync}|$ robots are involved in the S-GLBS and $|R_{est}|$ of these robots are active during the `extinguish fire` behavior,

where $|R_{est}| < |R_{sync}| < |R_{async}| < N$.

4.2 The Experiment Timeline

The experiment timeline (see Fig. 2) consists of a sequence of three swarm behaviors: (i) `detect fire`, (ii) `navigate toward fire`, and (iii) `extinguish fire`, as described in the following.

Detect Fire Behavior. The experiment starts at time t_0 when the swarm of N robots begins exploring the arena through a random walk, with the goal of locating the region affected by the fire (`detect fire` behavior in Fig. 2). At time

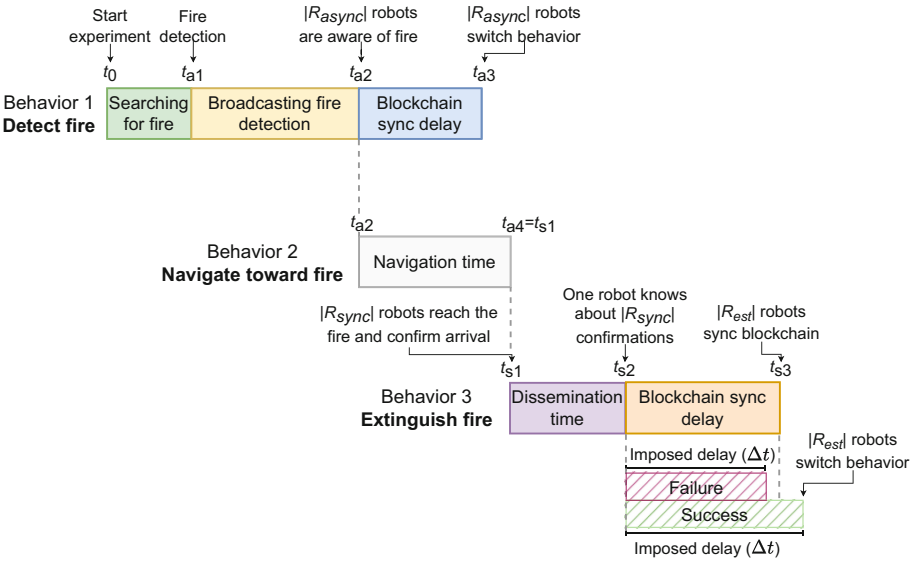

Fig. 2. Timeline of execution of the three behaviors used by the robot swarm in the fire extinguishing scenario—see Sect. 4.2 for more details.

t_{a1} (subscript a stands for asynchronous), for the first time one of the robots identifies the presence of a fire and sends a transaction—that is, the transaction is sent to a pool of unconfirmed transactions, as described in Sect. 3—containing information about the presence of a fire and its location (this transaction will be later added to the blockchain). The remaining robots become aware of the fire by synchronizing with the blockchain through peers when they are in communication proximity (i.e., they are at a distance of 40 cm) during random walk. When a robot receives information regarding fire detection, it confirms reception by sending a transaction, which activates the A-GLBS. The robots that are already aware of the fire continue the random walk to favor information dissemination until the smart contract has received $|R_{async}|$ confirmations (at t_{a2}). At this point, the robots in R_{async} get informed (by querying the smart contract) that it is time to switch their behavior from `fire detection` to `navigate toward fire`. The robots in R_{async} that first receive the blockchain version containing all the $|R_{async}|$ confirmations switch their behavior instantly at t_{a2}. The remaining robots in R_{async} switch their behavior later, as soon as they synchronize with the blockchain of the robots that have already switched their behavior. At time t_{a3} all the robots in R_{async} have switched their behavior and the A-GLBS aret_{a3}, therefore completed. This synchronization delay (from t_{a2} to t_{a3}) is due to the blockchain synchronization delay inherent to blockchain technology.

Navigate Toward Fire Behavior. During the `navigate toward fire` behavior, the robots in R_{async} move toward the area affected by the fire. The transition from `detect fire` to `navigate toward fire` happens gradually: the first robot (r_x) switches at t_{a2} and the others up to time t_{a3}. Navigation time is the time interval between t_{a2} and t_{a4}, i.e., the time elapsed from the moment the first robot $r_x \in R_{async}$ switches its behavior until $|R_{sync}|$ robots reach the fire location.

Extinguish Fire Behavior. At time $t_{s1} = t_{a4}$ (subscript s stands for synchronous), at least $|R_{sync}|$ robots have reached the fire and they have all sent their transactions confirming their arrival. Between t_{s1} and t_{s2}, called dissemination time, the blocks containing these transactions are generated and disseminated by the robots until, at time t_{s2}, at least one robot r_y has received the blockchain version containing all the $|R_{sync}|$ confirmations. Therefore, at t_{s2}, one of the robots would in principle be ready to start extinguishing the fire because it knows that the required number of robots are positioned around the fire. However, as we want the robots to start extinguishing the fire at the same time, it is necessary that they wait until at least $|R_{est}|$ of the robots in R_{sync} have the same version of the blockchain as r_y. This happens at time t_{s3}. Unfortunately, as the robots that synchronize their blockchain with the blockchain of robot r_y do not know the value of t_{s3}, they cannot decide how long to wait before starting to extinguish the fire. Our solution to achieve synchronized fire extinguishing is to impose a delay with respect to t_{s2} for switching to the extinguish fire behavior that needs to be large enough so that the S-GLBS is activated at or after t_{s3}. When selecting the value Δt to give to this imposed delay, one should consider that there is a trade-off between the success rate of the S-GLBS and its duration:

Table 1. Variables and parameters used in experiments

Variables	**Value**
Arena size (m^2)	7, 14, 28
Density (robots/m^2)	1, 2, 3, 4, 5, 6
Δt (s)	2.0, 2.1, 2.2, ..., 29.9, 30.0
Parameters	**Value**
R_{est}	4
ΔR_{sync}	2
ΔR_{async}	1
Fire patch diameter (cm)	60
Communication range (cm)	40
Block interval period (s)	2

higher values of Δt will increase both the S-GLBS duration and its probability of success, whereas lower values will make the S-GLBS faster but at the cost of a lower probability of success.

4.3 Experimental Setup

All experiments are conducted using the ARGoS [24] robot simulator, in which we have designed our fire extinguishing scenario (discussed in Sec. 4.1). As a robot model, we use the e-puck robot plugin, available in ARGoS. The e-puck robot [18] is equipped with a total of eight proximity sensors, used to detect and avoid obstacles as well as other robots. Additionally, we use the three ground sensors of the robot to simulate fire detection (black ground color indicates the presence of fire in Fig. 1). To estimate the presence of neighboring robots, the e-puck robot uses a range-and-bearing sensor. Finally, the robot is equipped with a pair of wheels, enabling locomotion and environmental exploration.

Every single robot is programmed to function as a node in the Ethereum blockchain by using the ARGoS-blockchain interface [30,31]. Robots communicate using range-and-bearing sensors only when they are in line-of-sight and the spatial distance between them does not exceed 40 cm. In each experiment, the number of robots is given by the product of the arena size times the swarm density (see Table 1). This yields experiments with up to 168 robots. A total of 30 iterations of each experiment are performed for every combination of density, imposed delay and arena size, as indicated in Table 1. The remaining experimental parameters are provided in Table 1.

5 Results

In this section, we analyze the performance of the two GLBS mechanisms considered in this paper, A-GLBS and S-GLBS, in terms of time needed to perform the switch and of success rate. The switch time is defined as follows:

– A-GLBS time is the time interval between t_{a1} (at least one robot detects the fire) and t_{a3} (the $|R_{async}|$ robots have switched their behavior); and
– S-GLBS time is the sum of the dissemination time $t_{s1} \rightarrow t_{s2}$ and the imposed delay Δt (see Fig. 2). For an experiment to be successful, S-GLBS time must be larger or equal to $t_{s1} \rightarrow t_{s3}$ (i.e., the blockchain synchronization delay must be smaller than the imposed delay).

We discuss the performance of A-GLBS in terms of A-GLBS time in Sect. 5.1, where we specifically study the effect of changing the swarm density and the arena size. In Sect. 5.2, we discuss the performance of S-GLBS in terms of time and success rate of the experiment. As explained in Sect. 4.2, the S-GLBS performance is expected to strongly depend on the imposed delay (Δt). For this reason, the imposed delay will be the main factor we consider while discussing these results, along with the density and the arena size.

5.1 Asynchronous Behavioral Switch (A-GLBS)

A-GLBS time is shown in Fig. 3 for the different arena sizes considered and for different swarm densities. The A-GLBS requires the dissemination of the information about fire detection and reaching consensus on this information on the blockchain. We expect that an increase of swarm density causes a decrease of A-GLBS time because the robots are likely to have a higher connectivity and can therefore disseminate information faster. We can observe that this expectation is met in all the different arena sizes considered (see Fig. 3). In addition, we observe that for all arena sizes variability tends to decrease for increasing densities. This is because when the density increases the probability of having communication delays caused by disconnected robots decreases and therefore the time required to complete the A-GLBS is less variable.

5.2 Synchronous Behavioral Switch (S-GLBS)

To analyze the performance of the S-GLBS, first we note that we consider the S-GLBS successful only if the blockchain synchronization (which completes at t_{s3},

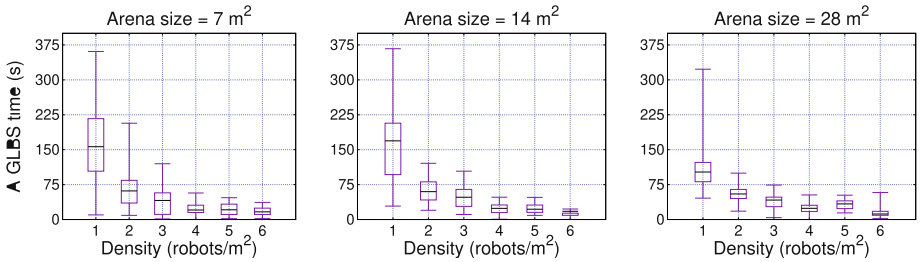

Fig. 3. Asynchronous GLBS: A-GLBS time ($t_{a1} \rightarrow t_{a3}$) as a function of the swarm density. Experiments are repeated 30 times. In box-plots, whiskers indicate the minimum and maximum values and horizontal black line indicates the median.

see Fig. 2) occurs before the end of the imposed delay Δt. In Fig. 4, we analyze the success rate in terms of percentage of failed experiments (top row) and S-GLBS time (middle and bottom row) as a function of Δt for varying swarm densities and arena sizes. When reporting the value of S-GLBS time corresponding to each imposed delay, we only considered the successful runs.

Results presented in Fig. 4 (first row) show that when the imposed delay increases the number of failed experiments decreases. However, this comes at the cost of an increased S-GLBS time (last two rows).

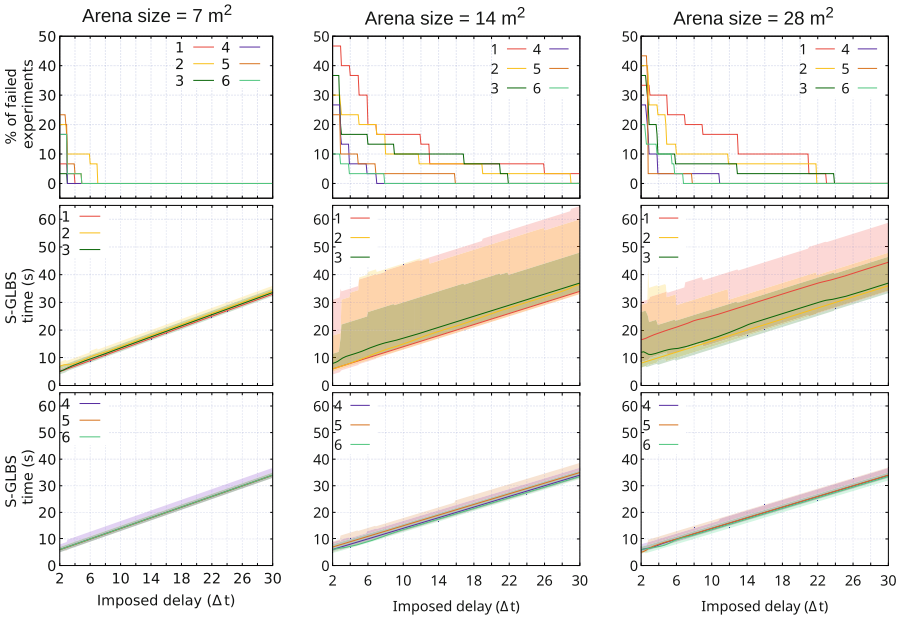

Fig. 4. Synchronous GLBS: Percentage of failed experiments (top row) and S-GLBS time (middle and bottom rows) as a function of imposed delay Δt. Experiments are repeated 30 times; S-GLBS time is computed only for successful experiments. The top row plots the percentage of failed experiments for six different swarm densities, the middle and bottom rows plot S-GLBS time for densities (robots/m^2) 1, 2, 3 and densities 4, 5, 6, respectively. In the middle and bottom rows, the lines represent the median and the shaded areas depict the inter-quartile range. Results show that in the small arena or when the density is $\geq$4 robots/m^2 there is no significant difference in S-GLBS time.

We also observe significant variations in the S-GLBS time for lower densities ($\leq$3 robots/m^2) in medium (14 m^2) and large (28 m^2) arenas as depicted in the middle row of Fig. 4. On the other hand, in the small arena (7 m^2) the degree of variation is lower. This can be explained as follows. The time required for the S-GLBS is influenced by two key factors: the block interval period and the block

propagation time (because transactions that contain the confirmations about the robots' arrivals at the fire need to be included in blocks and propagated among the robots). Because the number $|R_{sync}|$ of robots required for the S-GLBS does not change across arena sizes, it represents a larger proportion of the swarm in the small arena than in the larger ones. Therefore, it is highly likely that, in the small arena, either one of the robots $\in R_{sync}$ or one of the robots within the communication range (note that the communication range is constant across arena sizes but leads to a higher connectivity in the small arena) is capable of acting as a sealer for the PoA consensus protocol (see Sect. 3). This leads to a block production time with low variability. Hence, the variability of S-GLBS time is lower for the smaller arena. In contrast, when considering medium and large arenas, the number $|R_{sync}|$ of robots no longer constitutes a relatively significant proportion of the robot swarm. As a result, it is possible that there are longer delays before the blockchain transactions created by the robots $\in R_{sync}$ are incorporated into a block.

Conversely, in case of higher densities ($\geq$4 robots/m^2) (see bottom row of Fig. 4), enhanced connectivity among robots in the swarm facilitates the finding of an appropriate sealer as well as the efficient propagation of blocks. This leads to a lower degree of variability in S-GLBS time.

6 Conclusions and Future Work

This paper has presented the notion of group-level behavioral switch (GLBS) and its categorization into two distinct types: the asynchronous GLBS and the synchronous GLBS. In order to let a robot swarm reach consensus for a GLBS, a solution has been proposed based on blockchain technology, developed on the Ethereum platform, and evaluated using a fire extinguishing scenario as testbed. The results demonstrate that the time required for the A-GLBS is influenced by the density of the swarm: the higher the density, the lower the A-GLBS time. We also assessed the time required for the S-GLBS and its success rate as a function of the imposed delay Δt. The findings indicate that there exists a trade-off between the S-GLBS success rate and its time for completion. Our proposed solution is versatile, as it allows a user to choose the imposed delay depending on specific requirements of the given scenario. It should however be noted that the choice of the imposed delay in principle depends also on the underlying communication infrastructure which influences the speed with which the robots in the swarm synchronize with the most recent blockchain. In future work, we will investigate the exact form of this dependency as well as other approaches for synchronizing the behavioral switch without imposing the delay externally. Furthermore, in the presented research, we assumed that all robots in the swarm use a shared clock in order to achieve synchronicity in the case of S-GLBS. However, in our future work, we want to accomplish synchronization without relying on a common clock. We also intend to study the effects that the presence of Byzantine robots have on decision-making for GLBSs.

Acknowledgements. V. Strobel and M. Dorigo acknowledge support from the Belgian F.R.S.-FNRS, of which they are a Postdoctoral Researcher and a Research Director respectively.

References

1. Bahçeci, E., Şahin, E.: Evolving aggregation behaviors for swarm robotic systems: a systematic case study. In: Proceedings IEEE Swarm Intelligence Symposium, pp. 333–340 (2005)
2. Bahçeci, E., Soysal, O., Şahin, E.: A review: pattern formation and adaptation in multi-robot systems. Technical report. CMU-RI-TR-03-43, Carnegie Mellon University, Pittsburgh, PA (2003)
3. Bartashevich, P., Mostaghim, S.: Multi-featured collective perception with evidence theory: tackling spatial correlations. Swarm Intell. **15**, 1–28 (2021). https://doi.org/10.1007/s11721-021-00192-8
4. Brambilla, M., Ferrante, E., Birattari, M., Dorigo, M.: Swarm robotics: a review from the swarm engineering perspective. Swarm Intell. **7**(1), 1–41 (2013). https://doi.org/10.1007/s11721-012-0075-2
5. Buterin, V.: A next-generation smart contract and decentralized application platform. Ethereum project white paper. Technical report, Ethereum Foundation (2014). https://ethereum.org/en/whitepaper/. Accessed 02 July 2024
6. Chen, J., Sun, R., Kress-Gazit, H.: Distributed control of robotic swarms from reactive high-level specifications. In: 2021 IEEE 17th International Conference on Automation Science and Engineering (CASE), pp. 1247–1254 (2021). https://doi.org/10.1109/CASE49439.2021.9551578
7. Dorigo, M., Birattari, M., Brambilla, M.: Swarm robotics. Scholarpedia **9**(1), 1463 (2014). https://doi.org/10.4249/scholarpedia.1463
8. Dorigo, M., Pacheco, A., Reina, A., Strobel, V.: Blockchain technology for mobile multi-robot systems. Nat. Rev. Electr. Eng. **1**(4), 264–274 (2024). https://doi.org/10.1038/s44287-024-00034-9
9. Dorigo, M., Théraulaz, G., Trianni, V.: Reflections on the future of swarm robotics. Sci. Robot. **5**(49) (2020). https://doi.org/10.1126/scirobotics.abe4385
10. Dorigo, M., Théraulaz, G., Trianni, V.: Swarm robotics: past, present and future. Proc. IEEE **109**(7), 1152–1165 (2021). https://doi.org/10.1109/JPROC.2021.3072740
11. Ducatelle, F., Di Caro, G., Pinciroli, C., Gambardella, L.M.: Self-organized cooperation between robotic swarms. Swarm Intell. **5**, 73–96 (2011). https://doi.org/10.1007/s11721-011-0053-0
12. Ebert, J.T., Gauci, M., Mallmann-Trenn, F., Nagpal, R.: Bayes bots: collective Bayesian decision-making in decentralized robot swarms. In: 2020 IEEE International Conference on Robotics and Automation (ICRA), pp. 7186–7192 (2020). https://doi.org/10.1109/ICRA40945.2020.9196584
13. Ferrante, E., Turgut, A.E., Mathews, N., Birattari, M., Dorigo, M.: Flocking in stationary and non-stationary environments: a novel communication strategy for heading alignment. In: Schaefer, R., Cotta, C., Kołodziej, J., Rudolph, G. (eds.) PPSN 2010. LNCS, vol. 6239, pp. 331–340. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15871-1_34
14. Groß, R., Bonani, M., Mondada, F., Dorigo, M.: Autonomous self-assembly in swarm-bots. IEEE Trans. Rob. **22**(6), 1115–1130 (2006). https://doi.org/10.1109/TRO.2006.882919

15. Groß, R., Dorigo, M.: Self-assembly at the macroscopic scale. Proc. IEEE **96**(9), 1490–1508 (2008)
16. Groß, R., Dorigo, M.: Towards group transport by swarms of robots. Int. J. Bio-Inspired Comput. **1**(1–2), 1–13 (2009). https://doi.org/10.1504/IJBIC.2009.022770
17. Keramat, F., Peña Queralta, J., Westerlund, T.: Partition-tolerant and Byzantine-tolerant decision making for distributed robotic systems with IOTA and ROS2. IEEE Internet Things J. **10**(14), 12985–12998 (2023). https://doi.org/10.1109/JIOT.2023.3257984
18. Mondada, F., et al.: The e-puck, a robot designed for education in engineering. In: Gonçalves, P.J.S., et al. (eds.) Proceedings of the 9th Conference on Autonomous Robot Systems and Competitions, pp. 59–65. IPCB: Instituto Politècnico de Castelo Branco, Portugal (2009)
19. Nakamoto, S.: Bitcoin: a peer-to-peer electronic cash system (2008)
20. Nouyan, S., Campo, A., Dorigo, M.: Path formation in a robot swarm: self-organized strategies to find your way home. Swarm Intell. **2**(1), 1–23 (2008). https://doi.org/10.1007/s11721-007-0009-6
21. O'Grady, R., Groß, R., Christensen, A.L., Dorigo, M.: Self-assembly strategies in a group of autonomous mobile robots. Auton. Robot. **28**(4), 439–455 (2010). https://doi.org/10.1007/s10514-010-9177-0
22. Pacheco, A., Strobel, V., Dorigo, M.: A blockchain-controlled physical robot swarm communicating via an ad-hoc network. In: Dorigo, M., et al. (eds.) ANTS 2020. LNCS, vol. 12421, pp. 3–15. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-60376-2_1
23. Pacheco, A., Strobel, V., Reina, A., Dorigo, M.: Real-time coordination of a foraging robot swarm using blockchain smart contracts. In: Dorigo, M., et al. (eds.) ANTS 2022. LNCS, vol. 13491, pp. 196–208. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-20176-9_16
24. Pinciroli, C., et al.: ARGoS: a modular, parallel, multi-engine simulator for multi-robot systems. Swarm Intell. **6**(4), 271–295 (2012). https://doi.org/10.1007/s11721-012-0072-5
25. Pinciroli, C., Beltrame, G.: Swarm-oriented programming of distributed robot networks. Computer **49**, 32–41 (2016). https://doi.org/10.1109/MC.2016.376
26. Polge, J., Robert, J., Le Traon, Y.: Permissioned blockchain frameworks in the industry: a comparison. ICT Express **7**(2), 229–233 (2021). https://doi.org/10.1016/j.icte.2020.09.002
27. Rubenstein, M., Cornejo, A., Nagpal, R.: Programmable self-assembly in a thousand-robot swarm. Science **345**(6198), 795–799 (2014). https://doi.org/10.1126/science.1254295
28. St-Onge, D., Varadharajan, V.S., Švogor, I., Beltrame, G.: From design to deployment: decentralized coordination of heterogeneous robotic teams. Front. Robot. AI **7** (2020). https://doi.org/10.3389/frobt.2020.00051
29. Strobel, V., Castelló Ferrer, E., Dorigo, M.: Managing Byzantine robots via blockchain technology in a swarm robotics collective decision making scenario. In: Dastani, M., Sukthankar, G., André, E., Koenig, S. (eds.) Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems, pp. 541–549. AAMAS 2018. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC (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. Front. Robot. AI **7**(54) (2020). https://doi.org/10.3389/frobt.2020.00054

31. Strobel, V., Pacheco, A., Dorigo, M.: Robot swarms neutralize harmful Byzantine robots using a blockchain-based token economy. Sci. Robot. **8**(79), eabm4636 (2023). https://doi.org/10.1126/scirobotics.abm4636
32. Trianni, V., Nolfi, S., Dorigo, M.: Hole avoidance: experiments in coordinated motion on rough terrain. In: Groen, F., Amato, N., Bonarini, A., Yoshida, E., Kröse, B. (eds.) Intelligent Autonomous Systems 8 - IAS 8, pp. 29–36. IOS Press, Amsterdam (2004)
33. Valentini, G., Ferrante, E., Dorigo, M.: The best-of-n problem in robot swarms: formalization, state of the art, and novel perspectives. Front. Robot. AI **4**(9) (2017). https://doi.org/10.3389/frobt.2017.00009
34. Valentini, G., Ferrante, E., Hamann, H., Dorigo, M.: Collective decision with 100 kilobots: speed versus accuracy in binary discrimination problems. Auton. Agent. Multi-Agent Syst. **30**(3), 553–580 (2016). https://doi.org/10.1007/s10458-015-9323-3
35. Valentini, G., Hamann, H., Dorigo, M.: Self-organized collective decision making: the weighted voter model. In: Proceedings of 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2014), pp. 45–52. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC (2014)
36. Wood, G.: Ethereum: a secure decentralised generalised transaction ledger- EIP-150 revision. Technical report, Ethereum Foundation (2017)
37. Yang, G.Z., et al.: The grand challenges of science robotics. Sci. Robot. **3**(14) (2018). https://doi.org/10.1126/scirobotics.aar7650
38. Zhao, H., et al.: A generic framework for Byzantine-tolerant consensus achievement in robot swarms. In: Proceedings of the 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2023), pp. 8839–8846. IEEE (2023). https://doi.org/10.1109/IROS55552.2023.10341423
39. Zlot, R., Stentz, A., Dias, M.B., Thayer, S.: Multi-robot exploration controlled by a market economy. In: Proceedings of the 2002 IEEE International Conference on Robotics and Automation, pp. 3016–3023 (2002). https://doi.org/10.1109/ROBOT.2002.1013690