



Article

Energy-Efficient Blockchain-Enabled Multi-Robot Coordination for Information Gathering: Theory and Experiments [†]

Cesar E. Castellon ¹, Tamim Khatib ¹, Swapnoneel Roy ¹, Ayan Dutta ^{1,*} , O. Patrick Kreidl ¹
and Ladislau Bölöni ² 

¹ School of Computing, University of North Florida, Jacksonville, FL 32224, USA

² Department of Computer Science, University of Central Florida, Orlando, FL 32816, USA

* Correspondence: a.dutta@unf.edu; Tel.: +1-904-620-1313

[†] This paper is an extended version of our paper published in ICRA 2022, Philadelphia, PA, USA, 23–27 May 2022.

Abstract: In this work, we propose a blockchain-based solution for securing robot-to-robot communication for a task with a high socioeconomic impact—information gathering. The objective of the robots is to gather maximal information about an unknown ambient phenomenon such as soil humidity distribution in a field. More specifically, we use the proof-of-work (PoW) consensus protocol for the robots to securely coordinate while rejecting tampered data injected by a malicious entity. As the blockchain-based PoW protocol has a large energy footprint, we next employ an algorithmically-engineered energy-efficient version of PoW. Results show that our proposed energy-efficient PoW-based protocol can reduce energy consumption by 14% while easily scaling up to 10 robots.

Keywords: multi-robot system; blockchain; security; energy



Citation: Castellon, C.E.; Khatib, T.; Roy, S.; Dutta, A.; Kreidl, O.P.; Bölöni, L. Energy-Efficient Blockchain-Enabled Multi-Robot Coordination for Information Gathering: Theory and Experiments. *Electronics* **2023**, *12*, 4239. <https://doi.org/10.3390/electronics12204239>

Academic Editors: Mikolaj Karpinski, Oleksandr O. Kuznetsov and Roman Oliynykov

Received: 28 August 2023

Revised: 6 October 2023

Accepted: 10 October 2023

Published: 13 October 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Mobile robots are becoming a standard for information gathering from large geographic areas. Applications of this include data collection about the state of the crop for precision agriculture, current data sampling for ocean monitoring, and hotspot detection for search and rescue, among others [1–3]. In a classical information-gathering task, the robot is equipped with an information collection sensor, e.g., an NDVI camera, and a soil acidity measurement sensor, among others. The robot goes to k locations in the environment and uses its sensor to measure values at those locations. Given the robot has a limited onboard battery power supply, the robot cannot go to all the locations in the environment. Similarly, if the geographic area is large, one robot might not be enough, and multiple (n) robots need to be deployed that will coordinate among themselves while covering $n \cdot k$ locations. The objective is to infer the sensor measurements in the remaining unvisited locations conditioned on these collected $n \cdot k$ measurements. This is possible if the sensor measurements are correlated. For example, if one location $l_1 = (x_1, y_1)$ has the presence of weeds, then another location $l_2 = (x_2, y_2)$ will have high probabilities of having weeds if $\|l_1 - l_2\| < \mathbf{d}$, where \mathbf{d} is a positive constant. The robots should collect these measurements while coordinating (e.g., via communication) with each other during the collection process to decide the best places to collect the measurements from. These measurements are then used by the end users, e.g., farmers, to make more informed decisions about their applications.

Although this seems like an attractive toolkit for automated information gathering, this approach has some challenges as well. First, the optimal information gathering with n robots is shown to be NP-hard [4,5]. Secondly, these robots are often accumulated from untrusted sources for deployment. Therefore, like other cyber-physical systems,

these robots are vulnerable to cyber-attacks. Examples include denial of service, jamming, and data tampering attacks, among others. In this paper, we focus on one of these attacks, namely *data integrity attacks*. In these attacks, one or more malicious entities inject tampered or falsified information into the network. This might cause an irrecoverable negative socioeconomic impact. An example of this would be spraying herbicides on good crops and not on weeds, which would kill the crops. Currently, there is no standard for robot security, and the way to go is the hardware or software-based security mechanism. In this paper, we take the software route—we employ a blockchain-based proof-of-work (PoW) consensus protocol to secure the sensed data by the robots from being tampered with.

Standard blockchain-based schemes for crypto-currencies do not readily apply to multi-robot applications, since the robot network topology changes over time as the robots move around in the environment. Furthermore, one or more robots might be completely out of communication with the rest of the group for a considerable amount of time before regaining connectivity. Therefore, there is a need for a tailor-made blockchain-based PoW protocol to solve the multi-robot information-gathering problem. On the other hand, these security mechanisms are known to be power-hungry, and therefore, the robots might become non-operational if they run these security protocols on top of their prescribed sensing and computation routines. To this end, we employ an algorithmically-engineered blockchain-based PoW that reduces the energy requirement while guaranteeing the same level of security as the original protocol. This energy-efficient PoW version, which is premised upon an energy-optimized implementation of the SHA-256 encryption algorithm, has recently been published by us; the reader is referred to [6] for full details.

In this paper, we have tested our proposed energy-efficient blockchain-enabled multi-robot information-gathering technique with up to 10 simulated robots using MATLAB and Python 3. Experimental results show that robots can save up to 14% energy consumption with our energy-optimized version of SHA-256 used in place of the standard SHA-256. We further extrapolate our findings to more real-world scenarios involving a multi-robot system. We acknowledge that adding a layer of blockchain-based security protocol adds to the run times for decision-making and overall mission execution. However, it is a necessary step to make the data exchanged among the robots tamper-proof. Our results show that the amount of added time for this depends on the connectivity mechanism and the difficulty of the PoW consensus mechanism, among other factors. Figure 1 illustrates a sample multi-robot secure information-gathering scenario.

A preliminary version of this work appeared in ICRA 2022 [7]. We have extended the conference paper version mainly by employing an energy-efficient PoW protocol on the robots, whereas our conference paper assumed that the robots only have access to the traditional PoW protocol. Energy consumption of blockchain has been a major issue limiting its usage. While our preliminary results [7] show how blockchain can be useful in the context of robotics systems, the current work further complements it by making blockchain energy-efficient. The primary contributions of this paper are as follows:

1. This is the first study that integrates blockchain-based data security techniques against data tampering attempts into a multi-robot information-gathering framework under continuous, periodic, and opportunistic connectivity.
2. We employ an energy-efficient version of the blockchain-based proof-of-work (PoW) consensus protocol that is up to 14% more efficient than the original PoW implementation in terms of energy consumption.
3. Our proposed techniques in this paper study the security aspects in the multi-robot information-gathering problem setup from the novel perspectives of model estimation error, data vulnerability and its impact, and energy efficiency.



Figure 1. Illustration of a multi-robot information-gathering scenario. The links between the robots indicate the availability of communication. If all the links are present at every time, then the robots have continuous connectivity (CC). If all the links become available (or if the links create a connected network via a different topology) periodically, the robots then follow periodic connectivity (PC). On the other hand, if the white or the orange link is available, but there is no guarantee that all of them are available at any time, then we have opportunistic connectivity (OC). We want these communication protocols among the robots to be secure as well as energy-efficient.

The remainder of the paper is organized as the following. First, in Section 2, we discuss the state of the art in multi-robot information-gathering and security aspects in robotics, while comparing and contrasting our work in this paper against them. In Section 3, we summarize the information-gathering problem (i.e., models and assumptions) using the notation appearing later in the paper. Section 4 discusses the proposed secure communication algorithms and their energy-efficient versions are presented in Section 5. In Section 6, we present and discuss our experimental results and, finally, we conclude in Section 7.

2. Literature Review

Mobile robots can be used to autonomously gather meaningful information based on which future actions can be taken. Due to its sheer practical significance, the domain of information sensing using autonomous mobile robots has recently received considerable attention [1,8–16]. In this problem setup, the goal of the robot(s) is to plan paths of lengths k such that the maximum amount of information can be collected from the environment. The goal locations might or might not be decided in the beginning. In this paper, we study a setting that is more suited for lifelong monitoring—the robots are not given any specific goal locations. Instead, they can finish their exploration anywhere in the environment [9,12,17]. Unlike coverage path planning, where the robots have to go through all the locations in

the environment, here, the robots' goal would be to infer the sensor measurements at the locations that they have not visited and collected information from. It is also popular in the literature to assume that the robots have been given pre-defined goal locations, and their goal in that setting would be to plan k -length paths from the start to the goal locations while maximizing the amount of collected information [5,18–20]. Gaussian process (GP) regression is the most used information modeling and inference tool in information-gathering studies [21]. Following GP, information theoretic measures such as entropy or mutual information can be used to send the robots to the most informative locations in the environment [5,22,23]. Similar to our setting in this paper, many prior studies have started with dividing the environment into n sub-regions so that n robots can be uniquely assigned to them [5,22–24]. If the different parts of the environment have different information measures, the robots can communicate their findings, e.g., share their GP models and/or their sensor observations so far to “fuse” their inference models. This has been shown to perform better than using no such coordination [2,12]. For fusing the models, Gaussian mixture models with the Expectation Maximization algorithm [25] can be employed [2,12].

Advancements in multi-agent deep reinforcement learning (DRL) and its applications in robotics have also been applied to the problem of information gathering [26]. One of the first such works is by Said et al. [17], who used recurrent neural networks along with GP for information modeling with up to 10 robots. They have used a mean-field DRL [27] technique to effectively reduce the n -robot learning problem to a 2-agent learning problem. Wei and Zheng [28] proposed an independent learning technique with credit assignment to solve this notoriously difficult problem. Pan, Manjanna, and Hsieh [29] recently proposed a policy gradient-based DRL for multi-robot information sampling. Unlike the prior studies, they do not use GP as the underlying information inference tool. Viseras and Garcia [30] also proposed a DRL-based information-gathering technique for a multi-robot team that can exploit existing accurate information models.

Although communication is a costly operation in terms of energy consumption and robots' communication ranges (via WiFi, for example) are limited, most of the studies assume that the robots can maintain a continuously connected network among themselves so that data sharing is always possible. On the other hand, Dutta, Ghosh, and Kreidl [1] previously showed the computation-intensive nature of such maintenance algorithms. Maintaining periodic connectivity brings up another challenge: reconnection planning with a group of n mobile robots, even in a tree-like environment, is an NP-hard problem [31]. Under these circumstances, opportunistic connectivity is the go-to option. In this case, the robots are not required to maintain communication with others, but if two or more robots are within each other's communication ranges, they will form an ad hoc network to share data as required. Opportunistic connectivity has been shown to be effective in multi-robot tasks [9,10,32]. For a survey of available connectivity models and their applications in multi-robot systems, the reader is referred to [33].

The information collected by the robots might be sensitive, and therefore, protecting the integrity of such data is of the utmost importance. However, there is no standard security protocol for multi-robot coordination, although communication attacks have been approached from the point of view of fault diagnosis (e.g., [34–36]). One of the first works on blockchain-based PoW for protecting the data shared among the robots is by [37], where a swarm of robots is controlled using blockchain-based smart contracts. In our prior work [38], we also used PoW-based tamper-proof technology for multiple robots to collect information. Both of these studies assume a connected robot, unlike this paper, where other connectivity strategies such as periodic and opportunistic are also tested while the blockchain-based security protocol is enabled. PoW is one of the most popular consensus protocols in cryptocurrencies [39]. However, it is known to be significantly resource-intensive [40–42]. This poses a challenge in robotics, as the robots run on a limited onboard power supply. Proof-of-stake (PoS), another blockchain-based consensus protocol, has recently been shown to consume only a fraction of the energy required by

PoW (<https://bit.ly/3z1q3aS>). Which consensus protocol suits a multi-robot application the best has yet to be explored. For more information on consensus protocols, refer to [43].

3. Problem Setup

We have a set of n homogeneous robots $R = r_1, r_2, \dots, r_n$ that explore a shared environment. The environment is discretized into a planar graph $G_p = \{V, E\}$, where the node set V represents the information collection locations, and the connections among them are denoted by the edge set E . Each robot r_i has its unique sub-region for exploration, \mathcal{V}_i , and $\mathcal{V}_i \cap \mathcal{V}_j = \emptyset$. We have pre-calculated \mathcal{V}_i using K-medoids clustering [44]. An example is shown in Figure 2a. r_i is equipped with an on-board sensor which allows it to sense and collect information (e.g., radiation detector). The robots' observations are modeled to be noisy. A robot r_i starts from a node $v_i^0 \in \mathcal{V}_i$. The path or sequence of nodes $(v_i^0, v_i^1, v_i^2, \dots)$ that each robot r_i follows determines the sensed locations of the ambient environment \mathcal{Z} ; specifically, we denote by v_i^t the node that robot r_i enters at time step t and by $\mathcal{Z}(v_i^t)$ the associated (scalar, real-valued) measurement received by robot r_i .

We use a Gaussian process (GP) to model the uncertain environment and noisy measurement process. Let \mathbf{X} denote a Gaussian random vector of length $|V|$ with prior mean vector μ and covariance matrix Σ , where μ and Σ represent the (minimum mean-square-error) prediction over node set V and its corresponding uncertainty, respectively [21]. For any given $GP = (\mu, \Sigma)$, the volumetric measure of uncertainty is calculated by an information-theoretic metric, (differential) entropy, which is formally defined as $H(\mathbf{X}) = \frac{1}{2} \log |\Sigma| + \frac{|V|}{2} \log(2\pi e)$, where $|\Sigma|$ denotes the covariance matrix's determinant, while $|V|$ denotes the vertex set's cardinality. It is a standard assumption in kernel-based parameterizations of GPs that the correlation between two nodes is inversely proportional to the distances between them [4,10,21]. We exploit this property when computing entropy by approximating the computationally intensive matrix determinant $|\Sigma|$ by the product of the per-node variances (σ_v^2) along the diagonal of Σ . In turn, the associated entropy $H(\mathbf{X})$ decomposes additively across the nodes, with each per-node term given by:

$$H(X_v) = \frac{1}{2} \log(2\pi e \sigma_v^2). \tag{1}$$

These per-node entropies, with their sum (via the Hadamard inequality) serving as the upper bound for the true global entropy $H(\mathbf{X})$, drive the robots towards opportune locations for information collection. In each move cycle, the information value of past measurements is reflected in these entropies by virtue of optimal updates to the underlying GP statistics during each sensing cycle.

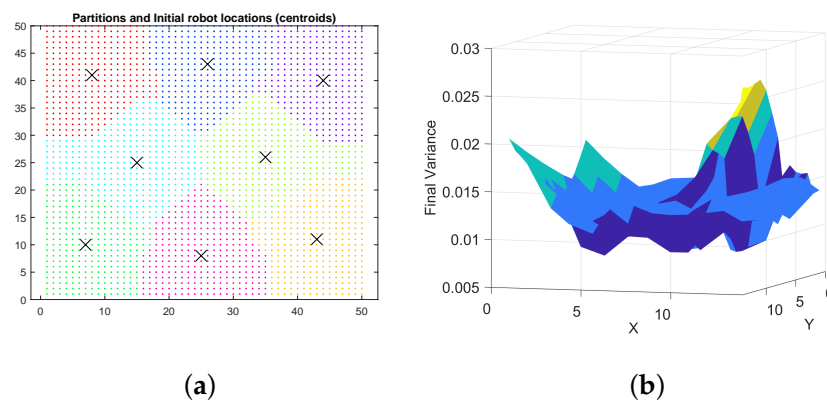


Figure 2. (a) A 50×50 grid environment is divided into eight non-overlapping sub-regions (shown using different colors) using K-medoids clustering. The centroids of these regions ('x') are the robots' initial positions. (b) An example of the final average variance map calculated by four robots in a 14×14 environment using a greedy strategy is shown, where the initial variance of the information model was 0.14. This shows the quality of the inference strategy.

Let us first summarize the sequential sense/move cycle of multi-robot information gathering in the context of initial time step 0. Assume each robot starts with a common initial GP model, called $GP = (\mu, \Sigma)$, and then takes measurement $\mathcal{Z}(v_i^0)$ at its start node $v_i^0 \in V$. Such prior statistics are typically derived pre-deployment from a training dataset and transferred onto each robot r_i before it is deployed to start node v_i^0 . We also assume the measurements are subject to additive white Gaussian noise $\epsilon \in \mathcal{N}(0, \sigma_n)$, in which case the updated local GP for robot r_i is given by the posterior statistics:

$$\begin{aligned} \Sigma_i^0 &= \Sigma - \Sigma \mathbf{C}v_i^{0'} [\mathbf{C}v_i^0 \Sigma \mathbf{C}v_i^{0'} + \sigma_n^2]^{-1} \mathbf{C}v_i^0 \Sigma \\ \mu_i^0 &= \mu + \Sigma_i^0 \mathbf{C}v_i^{0'} [\mathcal{Z}(v_i^0) - \mathbf{C}v_i^0 \mu], \end{aligned} \tag{2}$$

where $\mathbf{C}v_i^0$ denotes the length- $|V|$ row vector of all zeros except for a one in component v_i^0 , and $\mathbf{C}v_i^{0'}$ is its matrix transpose. During periodic or opportunistic connectivity, the posterior statistics will sometimes evolve on a batch of measurements, which is easily accommodated by appropriate augmentation of the output matrix $\mathbf{C}(\cdot)$; the reader is referred to [10] for more details. As the sensing step concludes, and each robot now possesses its updated GP statistics $GP_i^0 = (\mu_i^0, \Sigma_i^0)$ via Equation (2), the per-node rewards using Equation (1) are calculated. In a greedy fashion, robot r_i then chooses the next adjacent node $v_i^* \in \mathcal{V}_i$ that provides the (approximation of) maximum entropy:

$$v_i^* = \arg \max_{v \in \text{adj}(v_i^0)} H(v | GP_i^0) \quad \text{s.t. } v \in \mathcal{V}_i. \tag{3}$$

In the absence of inter-robot communication, each robot repeats the above sense-and-move cycle until it runs out of the given budget B . Such information-gathering strategies, interleaving GP-based sensor measurement processing with greedy entropy-based movement decisions, are well-studied in the literature and in certain conditions yield performance even provably bounded within constant factors of optimal [4,5,45]. In scenarios permitting inter-robot communication, connected robots during any cycle may also share actual measurements and/or GP statistics through which better-coordinated movement decisions become theoretically possible. Distributed multi-robot information gathering remains the subject of active research, with important considerations including questions of who talks to whom, how often, and how much, as well as how to integrate whatever information does get shared to ideally guarantee improved collective performance. The next section summarizes prior work in this area in the context of the blockchain-based security measures that the rest of this paper seeks to make more energy efficient.

4. Secure Communication Algorithms

Under periodic connectivity (PC) assumptions, the robots will form a connected network after every \mathcal{F} cycle, where \mathcal{F} refers to the coordination frequency [3,13]. (The reader seeking more details on how to reconnect the robots periodically is referred to [3,31].) Observe that the case of $\mathcal{F} = 1$ recovers the standard continuous connectivity (CC) assumptions. Opportunistic connectivity (OC) is distinctly different because the robots are not guaranteed to form connected communication networks; instead, the robots communicate only if and when at least two robots are within each other’s communication ranges. Note that, for connectivity discussions, we consider continuous connectivity (CC) as the baseline and, therefore, we first discuss blockchain’s proof-of-work (PoW) consensus for CC and discuss PC and OC thereafter.

4.1. Proof-of-Work (PoW) Consensus Protocol in CC

In an insecure version, each robot receives information from other robots and updates its local GP model using Equation (2) [1] (Algorithm 1). A malicious entity can attack this data-sharing system via data-tampering attempts [46,47]. To prevent other robots from incorporating such fake data for their future decision-making, we have used a blockchain-

based security protocol. Blockchain is a tamper-resistant digital ledger that the robots maintain in a distributed fashion [43]. In a blockchain, the data are stored in discrete units, called blocks, that are linked (chained) to each other by having the hash of one block as part of the data of the next block. Similar to [38], each robot r_i maintains a local blockchain C_i . Each block $b_{idx} \in C_i$ contains five primary components $\langle D, T, idx, N, H_{last} \rangle$, where D denotes the collected measurement(s), T represents the current timestamp, idx is the index of the block, N is an integer called nonce, and H_{last} represents the previous block b_{idx-1} 's hash.

Algorithm 1: Energy-efficient blockchain-enabled information gathering

```

1  $v_i^* \leftarrow r_i$ 's next location;
2 while  $budget \geq 0$  do
3   Go to  $v_i^*$  and gather information;
4   Create a block with these;
5   Share this block with the other robots (either periodically with PC or every cycle with
   CC or OC);
6   Receive blocks from other robots if applicable;
7   Decide whether the received information can be added to the local blockchain using the
   energy-efficient PoW (Algorithm 2);
8   Update the local GP with the new data using Equation (2) and recalculate entropy using
   Equation (1);
9   Decide  $v_i^*$  using Equation (3);

```

Algorithm 2 presents the pseudocode of the PoW algorithm. After r_i measures $\mathcal{Z}(v_i^*)$ at v_i^* , it puts them in D . The nonce is initially set to zero. The robot creates a block with it and finds its corresponding hash. To mine this block, r_i checks whether the hash has the required *difficulty* or not. In our implementation, a difficulty is first determined by counting the number of leading or trailing zeros in a block's hash value. Finding the hash value becomes harder the more zeros there are. The nonce is initialized to 0 and raised by 1 inside of a for loop—this is utilized to find this hash value. The loop terminates when the correct nonce is determined, i.e., the number of zeros matches the number of zeros in the hash value of the nonce variable. SHA-256 is used to calculate the hash value. A maximum loop number (MAX_STEPS, line 3 of Algorithm 2) is set. If the correct nonce cannot be identified by the maximum loop number, the nonce is reset to 0, and the software throws an error. *Mining* is the process of obtaining a correct nonce whose hash value satisfies the desired difficulty level. The specific details of different parameters are discussed next.

Algorithm 2: Proof-of-work (PoW) algorithm

```

Data: Block  $b$ , Nonce  $N$ , Difficulty  $d$ 
Result:  $b.hash, N$ 
1  $N \leftarrow 0$ ;
2  $diff \leftarrow fill("0", d)$ ; /* #0s correspond to difficulty */
3 while  $MAX\_STEPS$  not reached do
4    $b.hash \leftarrow$  Find the hash using SHA-256;
5   if  $diff$  is matched then
6      $break$ ; /* Found desired hash */
7   else
8     if  $i = 2^{256}$  then
9        $N \leftarrow 0$ ; /* Nonce  $N$  is reset */
10       $break$ ;
11  Increment  $N$  by 1 and retry;

```

Once this mining process is over, the block is placed into r_i 's local blockchain C_i . With CC, the robots share their newly created blocks with each other after every cycle of sense and measurement. The robots replace their local blockchains with the received blockchains if the blocks are validated and, as a result, at the end of each coordination cycle, every robot will have other robots' valid new blocks along with their existing blocks in their local blockchains [38] (Algorithm 2). Note that the verification of the hash is straightforward. A robot looks at the nonce in a particular block, finds its corresponding hash, and checks whether the hash has the desired difficulty level. If not, the block is rejected; otherwise, it is validated. Therefore, increasing the difficulty reduces the probability of it being compromised, while the time and energy required by the robots increase significantly.

4.2. PoW Consensus Protocol in PC and OC

With PC, r_i creates D with the last \mathcal{F} measurements. As the robots coordinate periodically, they do not get a chance to share their collected information every cycle. Therefore, each block will contain \mathcal{F} measurements in PC, whereas it contains only one in CC. The other components in the block are calculated in the same way as in CC. Having a larger block size has one advantage—the robots do not need to share information in every cycle, and therefore, the communication and mining overheads are significantly less. On the other hand, in a bandwidth-limited environment, sharing a larger block might be prohibitive. Furthermore, as the robots are not aware of others' collected data, the quality of their informative paths might be sub-par compared to CC.

With OC, when two or more robots $\bar{R} \subseteq R$ come within each other's communication ranges, they share their local blockchains, and the coordination happens in the same way as in CC. Each robot $r_i \in \bar{R}$'s local blockchain contains its observed data and any valid data it has received earlier from $r_j \in R$. As the robots are collecting data from disjoint sub-regions in the environment, they might have mutually exclusive local blockchains. This might lead to *orphan* blocks. An orphan block is a block that was mined and placed in the blockchain at some point. However, over time, a new blockchain was generated that did not include this block, leaving it abandoned. Orphan blocks only exist in OC. For example, suppose robot r_i has a local blockchain containing the following blocks $\{a, b, c, d\}$, and robot r_j has a local blockchain of $\{a, b, c, e, f, g\}$. Next, these two robots come within C distance. Following our algorithm, r_i will accept the longer blockchain of r_j , causing block d to be abandoned, namely, an orphan block. While block d , in particular, will no longer be used, the data within it will be extracted and put back into a memory buffer known as unconfirmed data that r_i maintains in OC for such scenarios. Note that this is *not* the same as block d ; the data D are the same, but the previous hash, the timestamp, and the nonce will all be different. Also, block d was still a valid block but was left out of the blockchain simply due to asynchronous coordination in OC and not because of malicious data. Although the data in block d are preserved, the block itself will stay orphaned, meaning the mining effort put into it is lost. Using our proposed algorithm, the robots will not lose any collected data. For proof of this statement, please see [7] (Lemma 2).

5. Optimizing Energy Consumption of the Robotics System

The blockchain-based solution for the integrity problem in robotics communication comes with the cost of energy consumption. Due to its distributed, decentralized, integrity-preserving, and auditable features, blockchain technology has recently been recognized as a crucial tool for solving network and cybersecurity issues [48–51]. Unfortunately, when applied, the blockchain's intricate workflow uses a lot of electricity (energy), defeating the purpose of many real-world applications while making it difficult to meet the demands of low-energy robotics applications. For instance, energy use in Bitcoin, a typical blockchain application, will soon exceed 7.67 GW annually, which is close to Austria's annual energy consumption (8.2 GW) [52]. Recent techniques for reducing energy consumption in blockchain include developing new blockchain-based system software [53–55], operating systems [56,57], hypervisors [16,58], and hardware [59,60]. While these approaches have

yielded significant improvements in reducing energy consumption levels in blockchain, to the best of our knowledge, the solutions do not approach the problem from an algorithmic perspective. In this work, we employ an energy-efficient version of PoW proposed in [6], albeit re-engineered for robotic networks. An illustration is shown in Figure 3.

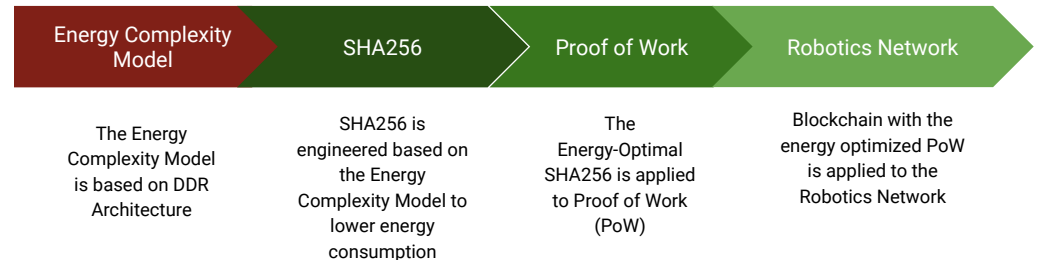


Figure 3. Our approach toward having an energy-optimized PoW for multi-robot communication [6].

Figure 3 summarizes our flow of actions. The energy complexity model (ECM) [61] yields the maximum savings in energy consumption (since it is the only process running on which ECM is applied) when applied to SHA-256. The energy-optimized SHA-256 when applied to PoW yields a lesser decrement in energy consumption levels. This is because the PoW algorithm’s computations other than SHA-256 add to the overall energy overhead. This is further evident when incorporated into the blockchain-based robotics network [7].

5.1. The Energy Complexity Model (ECM)

The energy complexity model (ECM), which has been applied to SHA-256 in this work, uses as its reference architecture the Double Data Rate Synchronous Dynamic Random Access Memory (DDR SDRAM). As illustrated in Figure 4, the main memory of DDR is divided into banks containing a fixed number of chunks. Although DDR specifications use the term *block*, we prefer the term *chunk* in the context of SHA-256 to avoid ambiguity. Data are distributed in chunks in each bank. Every bank also has a unique chunk known as the *sense amplifier*. Every data access requires bringing the chunk containing the desired data inside the sense amplifier of the appropriate bank. The current chunk must be returned to its bank before a fresh one can be brought in for the next access, since each sense amplifier can only contain one chunk at a time. As each bank has its sense amplifier, only one chunk of each bank can be accessed at once; however, chunks from various banks can be accessed simultaneously. Therefore, for a P bank DDR memory, we can always access P chunks. In the DDR3 version of the DDR architecture, the sensing amplifier is referred to as the per-bank cache.

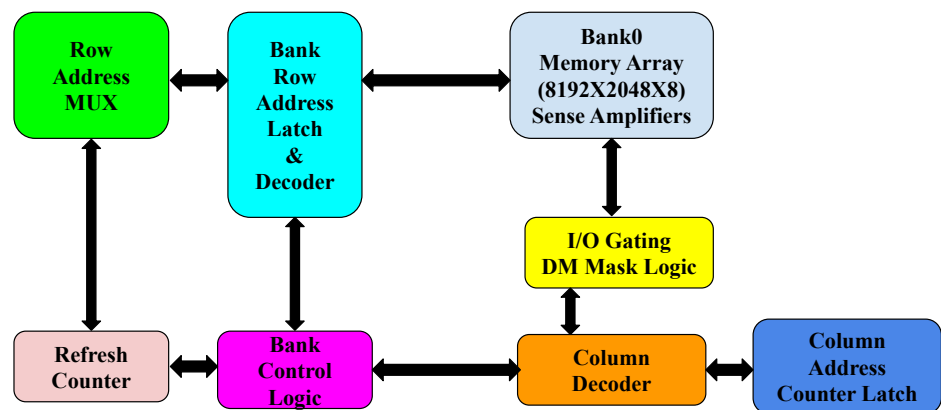


Figure 4. Block diagram of an internal DDR SDRAM memory chip.

A given DDR3 SDRAM’s P banks are identified by the ECM as M_1, M_2, \dots, M_P . Each bank M_i contains a cache C_i and several chunks of size B (in bytes). Figure 5 shows an example with $P = 4$ banks and only 4 chunks per bank. The chunks were given labels with the values $1, 2, \dots, 16$. The access patterns $(1, 2, 3, 4)$ or $(5, 6, 7, 8)$ are examples of totally serial execution given the restriction in DDR that only one chunk may be placed inside a certain cache C_i at any one time, whereas $(1, 5, 9, 13)$ or $(3, 8, 10, 13)$ are examples of completely parallel execution.

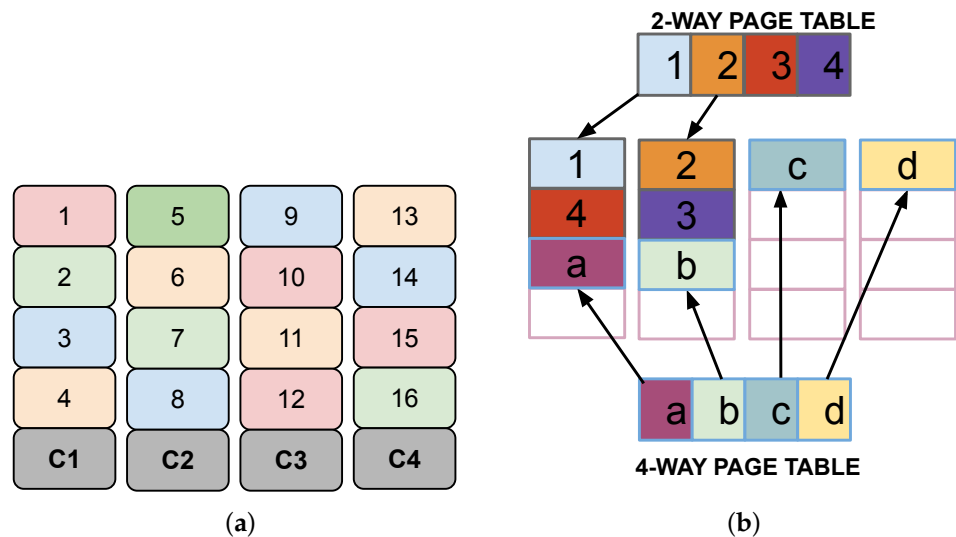


Figure 5. (a) An example of ECM for DDR3 with $P = 4$ banks; (b) Memory layout for different parallelization levels with $P = 4$.

Technically, according to a [61]-derived formula, the amount of energy used by an algorithm \mathcal{A} with the execution time τ for a P bank DDR3 architecture with B bytes per chunk is provided by:

$$E(\mathcal{A}) = \tau + (P \times B) / I. \tag{4}$$

The so-called *parallelization index*, represented by I , is effectively the number of parallel block accesses done by \mathcal{A} across different memory banks for every P block access made overall. ECM states that an algorithm’s potential for energy savings is inversely proportional to the extent to which it can be built to parallelize memory accesses.

5.2. Engineering SHA-256 Algorithm Using ECM

In this work, the underlying PoW hash algorithm SHA-256 has been engineered to use less energy by basing it on ECM. Initially, we describe how any algorithm \mathcal{A} can be parallelized using ECM. Next, we show how SHA-256, the PoW’s underlying hash algorithm, is designed for parallelization via ECM.

5.2.1. Parallelizing an Arbitrary Algorithm \mathcal{A}

The most frequent memory access sequence made by algorithm \mathcal{A} during execution for a particular input is first noted. The vector created by this access sequence is then designed to have the required amount of parallelism by constructing a logical mapping over memory blocks that house the data that \mathcal{A} has accessed. The physical location of the input (chunks) in the memory is fixed and is managed by the DDR memory controller. However, for various levels of parallelization, the order of access over chunks varies. Each time, a separate page table vector, or \mathbf{V} , is framed to perform a different level of access parallelization over physical chunks. The order of access among chunks is defined by \mathbf{V} (Figure 5b).

The page table vector \mathbf{V} contains the pattern $(1, 2, 3, 4, \dots)$ for 1-way access and $(1, 5, 9, 13, \dots)$ for 4-way access. The pattern of the page table vector \mathbf{V} is then mapped to

the input's original physical places using a function. The function to construct an ordering among the chunks is presented in [6] (Algorithm 2). Based on how we wish to access the chunks (P -way would signify full parallel access), the ordering is determined. By selecting chunks with *jumps*, the page table is filled. Jumps of P are chosen for P -way access to guarantee that consecutive chunk accesses are in P distinct banks. In accordance with the aforementioned example, jumps of 1 ensure that 4 successive chunk accesses occur in the same bank (bank 1 of Figure 5) for $P = 1$. In contrast, jumps of 4 ensure that 4 successive chunk accesses lie in 4 different banks for $P = 4$ (banks 1 through 4 of Figure 5).

5.2.2. Parallelizing SHA-256

As seen in Figure 6a, the SHA-256 algorithm divides the input into fixed-size message blocks that are then delivered sequentially to different compression methods. This block sequence is recognized in accordance with the SHA-256 algorithm's access pattern, which we subject to engineering using the ECM. The SHA-256 input vector (Figure 6a) is pre-processed into a different vector by using [6] (Algorithm 2). After that, the mapping is kept in a page table to be used in later hash computations.

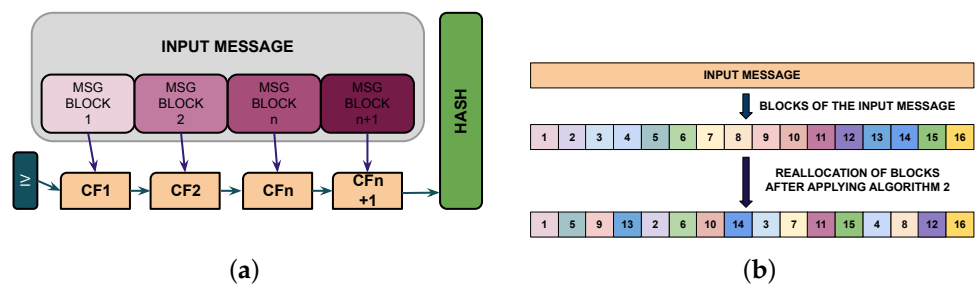


Figure 6. (a) An illustration of the original SHA-256 algorithm; (b) Access pattern re-engineering of blocks.

In Figure 6b, part of this procedure for 16 blocks and a parallelization index (jump) of 4 is displayed. The result of engineering the SHA-256 algorithm based on ECM is shown in Figure 7a. Note that the complexity of SHA-256 does not change because of this engineering [6] (Theorem 1). The parallelization index is set to $I = 8$, and an 8-bank DDR3 SDRAM is utilized in our experimentation. This basically means that we established a virtual mapping using the methods given in [61] to make sure that each set of 8 consecutive block access in SHA-256 occurs across all 8 banks.

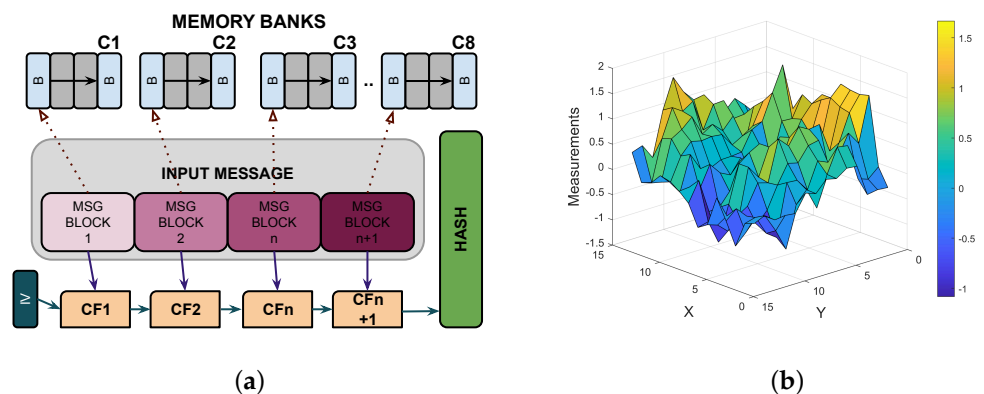


Figure 7. (a) An illustration of ECM-enhanced SHA-256; (b) The scalar information field used in the experiments.

5.3. Incorporating Energy-Optimized SHA-256 into PoW & Using Energy-Efficient PoW

Algorithm 2 employs SHA-256 in line 4 for PoW. Applying [6] (Algorithm 2) pre-processes the input vector in line 4 of the Algorithm 2, which is the concatenation of the

string (b.Params) and the nonce (line 4 of Algorithm 2). This is how the energy-optimized SHA-256 is used in our work, which is indicated by the green SHA-256 call in line 4. This energy-efficient PoW is then incorporated in this work for the robotics network for the cases of continuous (Section 4.1), periodic, and opportunistic connectivity (Section 4.2), respectively. To summarize, Algorithm 1 is executed in this work with the energy-efficient PoW algorithm for experiments wherever applicable.

6. Experiments

6.1. Setup and Results: Without Considering Energy Model

The parameters used in our experiments are listed in Table 1. The adversarial robot can inject fake data randomly sampled between $[-10, +10]$ in place of the original sensor measurement. The exact information field used in this paper is presented in Figure 7b.

Table 1. Parameters used in our experiments and their values.

Parameter	Value
Language	MATLAB and Python
Environment	Grid
Actions	8
Budget	20
Noise in sensing	$\mathcal{N}(0, 0.25)$
GP kernel	Exponential
Baselines	No Attack (no malicious data tampering) and Insecure (no security protocol in place)
for energy-efficient PoW	
RAM architecture	DDR3
Energy measurement software	pyRAPL [62]
OS	Linux Mint
Processor	Intel i5-2410M, 64-bit
C compiler	gcc 8.3.1

To first illustrate the effect of injecting fake data into a robotic information-gathering system and the impact of our secure technique to nullify that, we choose the mean-square-error (MSE) metric, which indicates how accurate the predicted information model is, to analyze the consequences of data integrity assaults on multi-robot information sampling. Figures 8 and 9a present the findings. The PC and OC results versus the results for CC [38] are also shown.

The PC version of MSE consistently outperforms the insecure version when statistically compared to its results. Similar to CC, the blockchain-based proposed solution will not be able to prevent efforts at data manipulation if the difficulty is set to a low value such as 1. The probability that the hash satisfies the difficulty 1 condition is $\frac{1}{16}$, which is relatively low, and as a result, the malicious robot can occasionally tamper with global data sharing. This is because there are 16 possible hash values per digit, and only one digit is an acceptable value for the prefix (0). The robots performed better—i.e., the final MSE was lower—when they communicated more frequently (for example, $\mathcal{F} = 2$ is better than $\mathcal{F} = 5$ when we compare the PC results with varying \mathcal{F}). Although this trend was constant, it is noteworthy that big differences in MSE were infrequently the result. We think that the reason there is such a modest variation in MSE outcomes across frequencies is that the robots that coordinate more frequently have more opportunities to modify their exploration plans (see Figure 9a for reference). Because of the aforementioned factor, it is often true that the connection model performs better in terms of MSE the closer it is to the CC. Be aware that this increases computing time, which we shall address in more detail later in the section.

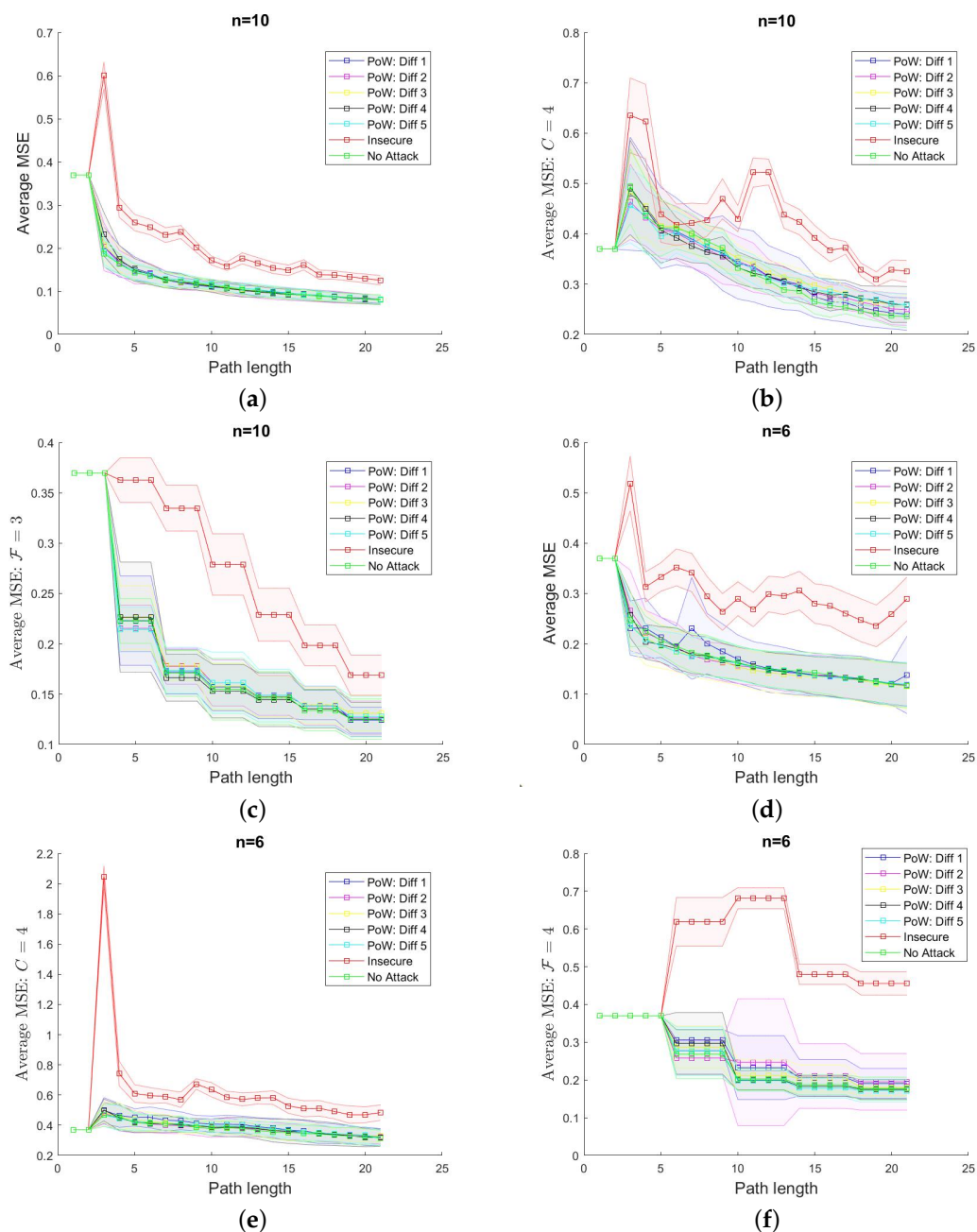


Figure 8. Single attacker: MSE comparison (the lower the better) among various connectivity models used: (a,d) CC with $n = 10$ and 6; (b,e) OC with $n = 10$ and 6; and (d,f) PC with $n = 10$ and 6.

Similar to CC and PC, the OC model almost always outperforms the insecure version statistically, with the exception of a few occasions where it performs less well (with difficulty 1) for the previously mentioned reasons. We have discovered that the MSE is lower with a bigger C compared to a smaller C . With different communication ranges, the MSE varies significantly. For instance, when the difficulty is 4 and n is set to 4, the final MSE value is 0.33 with $C = 4$ and 0.14 with $C = 12$. In almost every experiment, $C = 12$ performed statistically significantly better than $C = 4$. The MSEs with a range of 8 look more comparable to the range 4 than $C = 12$ with 2 robots. Because a third robot can still communicate with two robots that are out of range if there is one in range of the other two, communication range becomes less important as more robots are present. Therefore, the difference between having a range of 8 and 12 was significantly greater (up to 5.5 times

when n increases from 4 to 8 with difficulty 4). The robots needed less run time since they had less new data for PoW if they often interacted.

PC consistently beat CC in terms of algorithm run time (Figure 9b–d). Furthermore, when robots coordinate less frequently with PC, the run time is shorter. For instance, the run times for PC with $\mathcal{F} = 2$ and 5 are, respectively, 34.04 and 15.50 s, and the run time for CC is 59.76 s for the same $n = 10$ and difficulty 4.

On the other side, OC consistently outperformed CC but fell short of PC. Additionally, while OC typically performed better with a wider range, this is not always the case. This is due to the fact that, on some occasions, the time saved through coordinating less frequently was offset by the time required to redo PoW for the orphaned blocks.

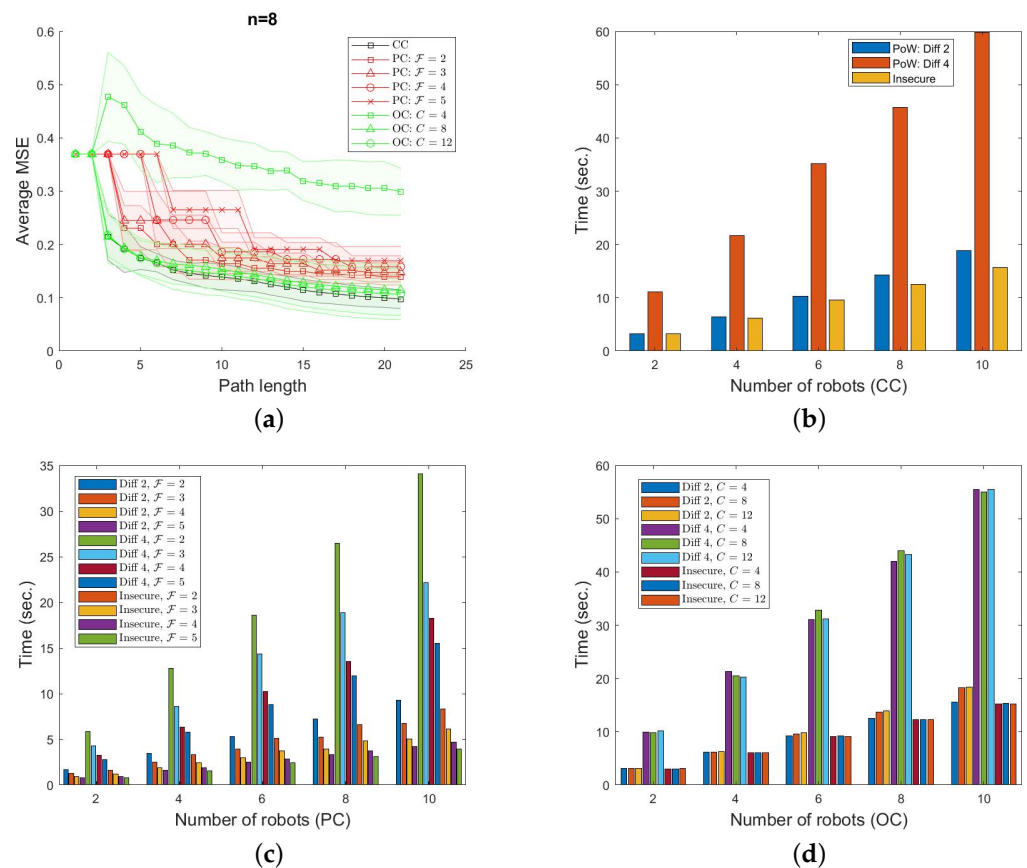


Figure 9. Single attacker: (a) comparison of MSE values among all the connectivity models with $n = 8$; run time comparison (the lower the better) between our proposed secure techniques and the implemented benchmark algorithms: (b) CC; (c) PC; and (d) OC.

6.2. Setup and Results: Considering the Energy Model

The specific parametric details for the energy-efficient PoW implementations are listed in Table 1. Remember that the ECM needs hardware with DDR RAMs.

6.2.1. Numeric Results

In our experiments, we performed three different lines of comparison for energy consumption.

1. We had the SHA-256 operation in PoW in our system implemented in two different programming languages, C and Python, for comparison. Furthermore, the energy-optimized SHA-256 was implemented only in C. Therefore, we had three different implementations for energy consumption comparison: (1) the standard implementation of SHA-256 using Python [P]; (2) the standard implementation of SHA-256 using

- C [S-C]; and (3) the engineered SHA-256 based on ECM for energy optimization using C [O-C].
2. We set three different difficulty levels for PoW (2, 3, and 4). Each difficulty level accounts for the number of leading 0s the generated hash needs to have to satisfy the condition in Algorithm 2. Next is the proof-of-work complexity index. As implied by Algorithm 2, a higher difficulty level accounts for more resource intensiveness in execution.
 3. We have also accounted for energy consumption (and optimization) for the three kinds of connectivity of robots, as illustrated in Figure 1. Robots have a continuous connection (CC) when all linkages are present at all times. Robots will then adhere to periodic connectivity (PC) if all links periodically become available (or if links connect to a network via a different topology). Opportunistic connectivity (OC) is what we have in contrast if either the white or orange link in Figure 1 is available, but there is no assurance that they will all be at any given time.

Figure 10 respectively compares the energy consumption of the robotics system using an energy-optimized and standard implementation of SHA-256 in the PoW algorithm. As mentioned before, we also add a standard Python implementation of SHA-256 in our experiments and measurements. The energy measurements of the Python implementation provide insights into how expensive Python is energy-wise compared to C for the same SHA-256 implementation. We have not engineered the Python implementation of SHA-256 for energy optimization. The energy-optimized version of SHA-256 consistently accounts for lower energy consumption as compared to the standard implementation of SHA-256 in C across Figure 10. The standard Python implementation of SHA-256, as expected, accounts for higher energy consumption than the standard C implementation of SHA-256. The only anomaly observed is in the opportunistic connectivity cases of Figure 10b,c. This can be explained by the randomness involved in opportunistic connectivity. The energy consumption in the case of opportunistic connectivity can be biased in experiments caused by the randomness involved.

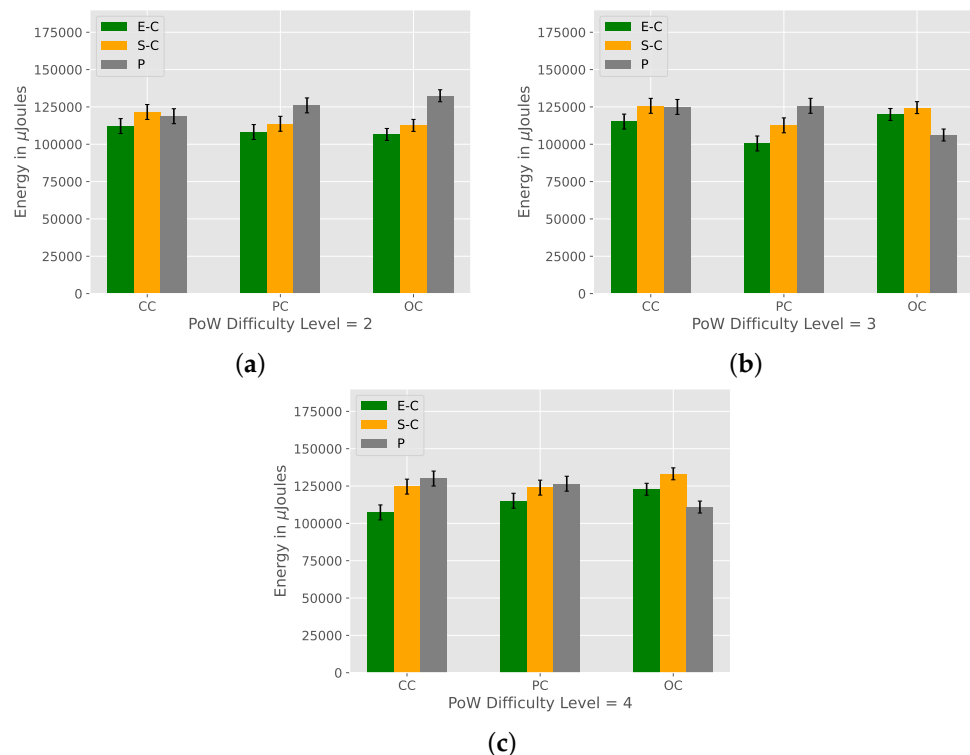


Figure 10. (a) C2, (b) C3, (c) C4 (with 1-sigma standard deviation over 500 trials).

Figure 11a summarizes the average energy savings across difficulty levels 2, 3, and 4 for different connectivity modes (CC, PC, and OC) in comparing the standard and energy-optimized versions of SHA-256 in C (the Python implementation has not been taken into account in Figure 11a). We observe energy savings up to 14% (in the case of CC with difficulty level 4). Section 6.2.2 provides an approximate estimation of energy savings in different similar real-world systems based on Figure 11a.

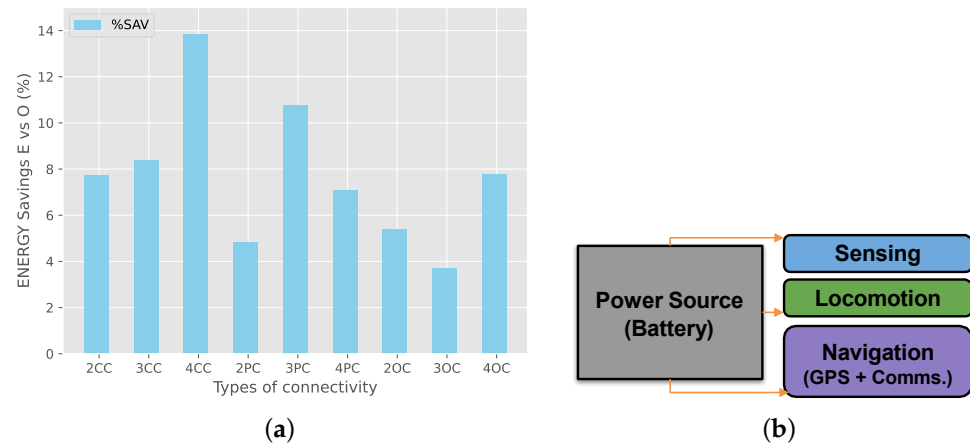


Figure 11. (a) Average energy savings (over 500 trials); (b) block diagram of energy consumption sources for unmanned devices.

6.2.2. Energy Savings Extrapolation over Real-World Systems

McNulty et al. in [63] establish the energy consumption for unmanned vehicle devices (UVD) to be divided into three subsystems: (1) navigation, (2) sensing, and (3) locomotion, as shown in Figure 11b. In our work, the communications module is part of the navigation subsystem. The total energy consumption of an unmanned device can therefore be expressed by $E_T = E_L + E_P + E_N$, where E_T stands for the total energy consumption, and E_L , E_P , and E_N stand for the energy consumed, respectively, by the locomotion, sensing, and navigation subsystems. Furthermore, $E_N = E_{gps} + E_{comm}$, where E_{gps} stands for energy consumed by the global positioning system and E_{comm} for the energy consumed by communications. Finally, $E_{comm} = E_{trx} + E_{sec}$, where E_{trx} stands for energy consumed by the transmission/reception process, and E_{sec} stands for energy consumed by the security tools implemented. E_{sec} will be equal to 0J for a system with no security. On the other hand, E_{sec} adds to the total energy consumption for security applications implemented within the system.

We provide an estimation of the distance a robot can cover in a full battery cycle for different kinds of robots in Table 2. In Table 2, the maximum distance a robot can cover (D_{max}) is for the case in which it has no security ($E_{sec} = 0$ J). The other two columns, D_{O-SHA} and D_{E-SHA} , estimate the distances covered when robots are implemented over the blockchain network using, respectively, the standard SHA and the energy-optimized SHA for PoW.

We used the following parameters from the robots' specification to estimate the amount of energy consumption per unit distance covered (we use the terms robot and UVD with the same meaning interchangeably in this section): (1) battery power capacity (BWh) expressed in watts per hour, (2) current output amp expressed in milli-amperes, (3) operating voltage vol expressed in volts, and (4) the maximum distance autonomy (D_{max}) expressed in meters. The total battery energy (TE) is calculated as [64]: $TE = amp \times vol \times 3600$. The energy a robot consumes to cover the unit distance (1 m), E_{unit} , is given by $E_{unit} = \frac{TE}{D_{max}}$. If we have security measures implemented for the robots (as a blockchain in our work), the distance covered by each robot over a full battery cycle will be less than D_{max} due to the energy consumed by security applications (E_{sec}). Let us call the distance a robot can go with security in place D_{sec} . Clearly, $D_{sec} < D_{max}$. The total distance cost, D_{cost} , which is the

distance that might have been covered with the energy consumed by security applications, can be estimated as $D_{cost} = \frac{E_{sec}}{E_{unit}}$. Finally, $D_{sec} = D_{max} - D_{cost}$.

In Table 2, we estimate D_{sec} when using the standard SHA-256, (D_{O-SHA}) and the energy-optimized SHA-256 (D_{E-SHA}) for the blockchain. To estimate E_{sec} in our work, since the blockchain is used as a security mechanism, we consider the energy consumption per block generated. A block is generated and added each time a robot communicates with another robot. A block generation and insertion in the blockchain involves a full cycle of Merkle tree generation and a subsequent PoW execution. Let us denote the energy consumption of each block operation (generation and insertion) by E_{block} . In [65], the authors estimated E_{block} to be, respectively, 2800 J and 2500 J for an input size of 256 B when standard SHA-256 and energy-optimized SHA-256 were used for the operation. Therefore, E_{sec} in our work can be estimated as a function of the number of block operations (B_{num}) performed by robots during communication: $E_{sec} = E_{block} \times B_{num}$. B_{num} for a specific robot depends on the number of times that the robot communicates (i.e., exchanges information) with other robots throughout D_{max} . In Table 2, we assume each robot to communicate once every 5 m. That is how E_{sec} has been calculated for the two cases (standard SHA-256 and energy-optimized SHA-256) and, in turn, the respective values of distances traveled by each robot during these two cases (D_{O-SHA} and D_{E-SHA}) have been estimated for the different kinds of robots listed in the table.

Table 2. Distance comparison for various types of hardware available for multi-robot implementation and research.

Type	Name	D_{max}	D_{O-SHA}	D_{E-SHA}
Aerial	DJI3	14,000	13,492	13,546
Aerial	Anafi Ai	32,640	27,101	27,695
Aerial	Bebop 3	9900	7359	7631
Aerial	Matrice RTK	75,900	51,008	53,675
Ground	TurtleBot4	7200	4960	5200
Ground	Jackal	28,800	27,725	27,840
Ground	Husky	10,800	10,498	10,530
Ground	TurtleBot Waffle Pi	1872	1773	1784

As depicted in Table 2, the use of blockchain as a solution to integrity problems in multi-robot communication is costly in terms of energy consumption and in terms of the distance covered by robots per battery cycle. The use of the energy-efficient SHA-256 in blockchain reduces the cost of energy while keeping the same level of security as the standard SHA-256.

7. Conclusions

In this paper, we study the problem of data collection from an unknown environment using a group of mobile robots. As opposed to traditional assumptions in this domain, our robots are vulnerable to cyber-attacks. In particular, we study data-tampering attempts, which can lead to unwanted actions taken by human operators. We have proposed a proof-of-work-based consensus protocol that has the foundation of a blockchain to secure the data shared among the robots. Although we have applied the proposed securing mechanism to a multi-robot information-gathering application, we believe our proposed technique can be used for numerous other applications where multi-robot communication is required for meaningful coordination. Furthermore, we have employed an energy-efficiency technique to reduce the energy footprint of PoW. Results show that our proposed security technique scales up to 10 robots while reducing the energy consumption of the employed PoW protocol by up to 14%. Unsurprisingly, our results show that adding a blockchain-based security mechanism adds to the overall execution time. This additional time is mostly notable in the case of continuous connectivity, and is least notable in the case of opportunistic. Although this added time is up to three-fold, this is necessary for

secure multi-robot coordination. Furthermore, we found that, even with 10 robots and a difficulty level of 4, the execution time was less than one minute—a relatively moderate number. This study proves that this is a promising direction of research in securing multi-robot coordination.

In the future, we plan to explore other avenues of data security that do not necessarily rely on blockchains to investigate whether it is possible to develop a more energy-efficient protocol while ensuring the security of the collected information. This will lead to further solutions to robotic communication problems with viable security and efficiency in terms of time and energy. We will also explore techniques leading to the detection of tampered data, such as in the case where intruders introduce tampered data before storing it in the blockchain, e.g., through sensor hardware tampering. Additionally, the current work generically simulates real-world robotics networks. Future research directions include implementing the blockchain network on real-world robotics networks to validate the simulation results presented in this paper.

Author Contributions: Conceptualization, all; methodology, C.E.C., S.R. and T.K.; software, C.E.C., S.R. and T.K.; validation, C.E.C., S.R. and T.K.; formal analysis, C.E.C. and S.R.; investigation, C.E.C. and S.R.; resources, S.R., A.D. and O.P.K.; data curation, C.E.C., S.R. and T.K.; writing—original draft preparation, all; writing—review and editing, all; visualization, C.E.C., S.R., T.K. and A.D.; supervision, S.R., A.D., O.P.K. and L.B.; project administration, S.R. and A.D.; funding acquisition, S.R., A.D., O.P.K. and L.B. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported in part by National Science Foundation (NSF) CPS Grants #1932300 and #1931767.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Dutta, A.; Ghosh, A.; Kreidl, O.P. Multi-robot Informative Path Planning with Continuous Connectivity Constraints. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 3245–3251.
2. Dutta, A.; Roy, S.; Kreidl, O.P.; Bölöni, L. Multi-robot information gathering for precision agriculture: Current state, scope, and challenges. *IEEE Access* **2021**, *9*, 161416–161430. [[CrossRef](#)]
3. Hollinger, G.A.; Singh, S. Multirobot coordination with periodic connectivity: Theory and experiments. *IEEE Trans. Robot.* **2012**, *28*, 967–973. [[CrossRef](#)]
4. Krause, A.; Singh, A.; Guestrin, C. Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies. *J. Mach. Learn. Res.* **2008**, *9*, 235–284.
5. Singh, A.; Krause, A.; Guestrin, C.; Kaiser, W.J. Efficient informative sensing using multiple robots. *J. Artif. Intell. Res.* **2009**, *34*, 707–755. [[CrossRef](#)]
6. Castellon, C.E.; Roy, S.; Kreidl, O.P.; Dutta, A.; Bölöni, L. Toward a Green Blockchain: Engineering Merkle Tree and Proof of Work for Energy Optimization. *IEEE Trans. Netw. Serv. Manag.* **2023**, *19*, 3847–3857.
7. Samman, T.; Dutta, A.; Kreidl, O.P.; Roy, S.; Bölöni, L. Secure Multi-Robot Information Sampling with Periodic and Opportunistic Connectivity. In Proceedings of the 2022 International Conference on Robotics and Automation (ICRA), Philadelphia, PA, USA, 23–27 May 2022; pp. 4951–4957.
8. Banfi, J.; Li, A.Q.; Rekleitis, I.; Amigoni, F.; Basilico, N. Strategies for coordinated multirobot exploration with recurrent connectivity constraints. *Auton. Robot.* **2018**, *42*, 875–894. [[CrossRef](#)]
9. Dutta, A.; Bhattacharya, A.; Kreidl, O.P.; Ghosh, A.; Dasgupta, P. Multi-robot informative path planning in unknown environments through continuous region partitioning. *Int. J. Adv. Robot. Syst.* **2020**, *17*, 1729881420970461. [[CrossRef](#)]
10. Dutta, A.; Patrick Kreidl, O.; O’Kane, J.M. Opportunistic multi-robot environmental sampling via decentralized markov decision processes. In *Distributed Autonomous Robotic Systems*; Springer: Cham, Switzerland, 2021; pp. 163–175.
11. Gao, C.; Ma, J.; Li, T.; Shen, Y. Hybrid swarm intelligent algorithm for multi-UAV formation reconfiguration. *Complex Intell. Syst.* **2023**, *9*, 1929–1962. [[CrossRef](#)]
12. Luo, W.; Sycara, K. Adaptive sampling and online learning in multi-robot sensor coverage with mixture of gaussian processes. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018; pp. 6359–6364.
13. Lauri, M.; Heinänen, E.; Frintrop, S. Multi-robot active information gathering with periodic communication. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; pp. 851–856.

14. Trejo, J.A.V.; Ponsart, J.C.; Adam-Medina, M.; Valencia-Palomo, G.; Theilliol, D. Distributed Observer-based Leader-following Consensus Control for LPV Multi-agent Systems: Application to multiple VTOL-UAVs Formation Control. In Proceedings of the 2023 International Conference on Unmanned Aircraft Systems (ICUAS), Warsaw, Poland, 6–9 June 2023; pp. 1316–1323.
15. Viseras, A.; Xu, Z.; Merino, L. Distributed multi-robot cooperation for information gathering under communication constraints. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018; pp. 1267–1272.
16. Zhang, S.; Lim, W.Y.B.; Ng, W.C.; Xiong, Z.; Niyato, D.; Shen, X.S.; Miao, C. Towards Green Metaverse Networking: Technologies, Advancements and Future Directions. *IEEE Netw.* **2023**, 1–10. [[CrossRef](#)]
17. Said, T.; Wolbert, J.; Khodadadeh, S.; Dutta, A.; Kreidl, O.P.; Bölöni, L.; Roy, S. Multi-robot information sampling using deep mean field reinforcement learning. In Proceedings of the 2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Melbourne, Australia, 17–20 October 2021; pp. 1215–1220.
18. Hollinger, G.A.; Sukhatme, G.S. Sampling-based robotic information gathering algorithms. *Int. J. Robot. Res.* **2014**, *33*, 1271–1287. [[CrossRef](#)]
19. Ma, K.C.; Ma, Z.; Liu, L.; Sukhatme, G.S. Multi-robot informative and adaptive planning for persistent environmental monitoring. In *Distributed Autonomous Robotic Systems*; Springer: Cham, Switzerland, 2018; pp. 285–298.
20. Wei, Y.; Zheng, R. Informative path planning for mobile sensing with reinforcement learning. In Proceedings of the IEEE INFOCOM 2020-IEEE Conference on Computer Communications, Toronto, ON, Canada, 6–9 July 2020; pp. 864–873.
21. Rasmussen, C.E. Gaussian processes in machine learning. In *Advanced Lectures on Machine Learning*; Springer: Berlin/Heidelberg, Germany, 2003; pp. 63–71.
22. Ma, K.C.; Liu, L.; Sukhatme, G.S. Informative planning and online learning with sparse gaussian processes. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; pp. 4292–4298.
23. Singh, A.; Krause, A.; Kaiser, W.J. Nonmyopic Adaptive Informative Path Planning for Multiple Robots. In Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI 2009), Pasadena, CA, USA, 11–17 July 2009; Bouilrier, C., Ed., 2009; pp. 1843–1850.
24. Kemna, S.; Caron, D.A.; Sukhatme, G.S. Adaptive informative sampling with autonomous underwater vehicles: Acoustic versus surface communications. In Proceedings of the OCEANS 2016 MTS/IEEE Monterey, Monterey, CA, USA, 19–23 September 2016; pp. 1–8.
25. Dempster, A.P.; Laird, N.M.; Rubin, D.B. Maximum likelihood from incomplete data via the EM algorithm. *J. R. Stat. Soc. Ser. B (Methodol.)* **1977**, *39*, 1–22. [[CrossRef](#)]
26. Orr, J.; Dutta, A. Multi-Agent Deep Reinforcement Learning for Multi-Robot Applications: A Survey. *Sensors* **2023**, *23*, 3625. [[CrossRef](#)] [[PubMed](#)]
27. Yang, Y.; Luo, R.; Li, M.; Zhou, M.; Zhang, W.; Wang, J. Mean Field Multi-Agent Reinforcement Learning. In Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholm, Sweden, 10–15 July 2018; Dy, J.G., Krause, A., Eds.; Volume 80, pp. 5567–5576.
28. Wei, Y.; Zheng, R. Multi-robot path planning for mobile sensing through deep reinforcement learning. In Proceedings of the IEEE INFOCOM 2021-IEEE Conference on Computer Communications, Vancouver, BC, Canada, 10–13 May 2021; pp. 1–10.
29. Pan, L.; Manjanna, S.; Hsieh, M.A. MARLAS: Multi Agent Reinforcement Learning for cooperated Adaptive Sampling. *arXiv* **2022**, arXiv:2207.07751.
30. Viseras, A.; Garcia, R. DeepIG: Multi-robot information gathering with deep reinforcement learning. *IEEE Robot. Autom. Lett.* **2019**, *4*, 3059–3066. [[CrossRef](#)]
31. Banfi, J.; Basilico, N.; Amigoni, F. Multirobot reconnection on graphs: Problem, complexity, and algorithms. *IEEE Trans. Robot.* **2018**, *34*, 1299–1314. [[CrossRef](#)]
32. Andre, T.; Bettstetter, C. Collaboration in multi-robot exploration: To meet or not to meet? *J. Intell. Robot. Syst.* **2016**, *82*, 325–337. [[CrossRef](#)]
33. Amigoni, F.; Banfi, J.; Basilico, N. Multirobot exploration of communication-restricted environments: A survey. *IEEE Intell. Syst.* **2017**, *32*, 48–57. [[CrossRef](#)]
34. Ghiasi, M.; Niknam, T.; Wang, Z.; Mehrandezh, M.; Dehghani, M.; Ghadimi, N. A comprehensive review of cyber-attacks and defense mechanisms for improving security in smart grid energy systems: Past, present and future. *Electr. Power Syst. Res.* **2023**, *215*, 108975. [[CrossRef](#)]
35. Xu, H.; Sun, Z.; Cao, Y.; Bilal, H. A data-driven approach for intrusion and anomaly detection using automated machine learning for the Internet of Things. *Soft Comput.* **2023**, *27*, 14469–14481. [[CrossRef](#)]
36. Zhou, H.; Zheng, Y.; Jia, X.; Shu, J. Collaborative prediction and detection of DDoS attacks in edge computing: A deep learning-based approach with distributed SDN. *Comput. Netw.* **2023**, *225*, 109642. [[CrossRef](#)]
37. 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* **2020**, *7*, 54. [[CrossRef](#)]

38. Samman, T.; Spearman, J.; Dutta, A.; Kreidl, O.P.; Roy, S.; Bölöni, L. Secure Multi-Robot Adaptive Information Sampling. In Proceedings of the 2021 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR), New York, NY, USA, 25–27 October 2021; pp. 125–131. [\[CrossRef\]](#)
39. Nakamoto, S. Bitcoin: A Peer-to-Peer Electronic Cash System. 2008. Available online: https://www.uscc.gov/sites/default/files/pdf/training/annual-national-training-seminar/2018/Emerging_Tech_Bitcoin_Crypto.pdf (accessed on 27 August 2023).
40. Dittmar, L.; Praktiknjo, A. Could Bitcoin emissions push global warming above 2 °C? *Nat. Clim. Chang.* **2019**, *9*, 656–657. [\[CrossRef\]](#)
41. Egíyi, M.A.; Ofoegbu, G.N. Cryptocurrency and climate change: An overview. *Int. J. Mech. Eng. Technol. (IJMET)* **2020**, *11*, 15–22.
42. Mora, C.; Rollins, R.L.; Taladay, K.; Kantar, M.B.; Chock, M.K.; Shimada, M.; Franklin, E.C. Bitcoin emissions alone could push global warming above 2 °C. *Nat. Clim. Chang.* **2018**, *8*, 931–933. [\[CrossRef\]](#)
43. Salimitari, M.; Chatterjee, M.; Fallah, Y.P. A survey on consensus methods in blockchain for resource-constrained IoT networks. *Internet Things* **2020**, *11*, 100212. [\[CrossRef\]](#)
44. Kaufmann, L. Clustering by means of medoids. In Proceedings of the Statistical Data Analysis Based on the L1-norm and Related Methods, Neuchatel, Switzerland, 31 August–4 September 1987; pp. 405–416.
45. Cao, N.; Low, K.H.; Dolan, J.M. Multi-robot informative path planning for active sensing of environmental phenomena: A tale of two algorithms. In Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems, Saint Paul, MN, USA, 6–10 May 2013; Gini, M.L., Shehory, O., Ito, T., Jonker, C.M., Eds.; pp. 7–14.
46. Gupta, M.; Abdelsalam, M.; Khorsandroo, S.; Mittal, S. Security and privacy in smart farming: Challenges and opportunities. *IEEE Access* **2020**, *8*, 34564–34584. [\[CrossRef\]](#)
47. Krishna, C.L.; Murphy, R.R. A review on cybersecurity vulnerabilities for unmanned aerial vehicles. In Proceedings of the 2017 IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR), Shanghai, China, 11–13 October 2017; pp. 194–199.
48. Issa, W.; Moustafa, N.; Turnbull, B.; Sohrabi, N.; Tari, Z. Blockchain-based federated learning for securing internet of things: A comprehensive survey. *ACM Comput. Surv.* **2023**, *55*, 191. [\[CrossRef\]](#)
49. Laghari, A.A.; Khan, A.A.; Alkanhel, R.; Elmannai, H.; Bourouis, S. Lightweight-BIoV: Blockchain Distributed Ledger Technology (BDLT) for Internet of Vehicles (IoVs). *Electronics* **2023**, *12*, 677. [\[CrossRef\]](#)
50. Rajasekar, V.; Sathya, K. Blockchain utility in renewable energy. In *Blockchain-Based Systems for the Modern Energy Grid*; Elsevier: Amsterdam, The Netherlands, 2023; pp. 115–134.
51. Gupta, S.S. *Blockchain for Secure Healthcare Using Internet of Medical Things (IoMT)*; Springer Nature: Cham, Switzerland, 2023.
52. Luo, H.; Liu, S.; Xu, S.; Luo, J. LECast: A Low-Energy-Consumption Broadcast Protocol for UAV Blockchain Networks. *Drones* **2023**, *7*, 76. [\[CrossRef\]](#)
53. Li, S.; Li, J.; Pei, J.; Wu, S.; Wang, S.; Cheng, L. Eco-CSAS: A Safe and Eco-Friendly Speed Advisory System for Autonomous Vehicle Platoon Using Consortium Blockchain. *IEEE Trans. Intell. Transp. Syst.* **2023**, *24*, 7802–7812. [\[CrossRef\]](#)
54. Oudani, M.; Sebbar, A.; Zkik, K.; El Harraki, I.; Belhadi, A. Green Blockchain based IoT for secured supply chain of hazardous materials. *Comput. Ind. Eng.* **2023**, *175*, 108814. [\[CrossRef\]](#)
55. Qi, L.; Tian, J.; Chai, M.; Cai, H. LightPoW: A trust based time-constrained PoW for blockchain in internet of things. *Comput. Netw.* **2023**, *220*, 109480. [\[CrossRef\]](#)
56. Gupta, M.; Patel, R.; Jain, S. Analysis of Blockchain Integration with Internet of Vehicles: Challenges, Motivation, and Recent Solution. In *Role of Data-Intensive Distributed Computing Systems in Designing Data Solutions*; Springer: Cham, Switzerland, 2023; pp. 129–163.
57. Praveena, B.; Reddy, P.V.P. Blockchain based Sensor System Design For Embedded IoT. *J. Comput. Inf. Syst.* **2023**, 1–18. [\[CrossRef\]](#)
58. Ali, M.H.; Jaber, M.M.; Khalil Abd, S.; Alkhayyat, A.; Q, M.R.; Ali, M.H. Application of internet of things-based efficient security solution for industrial. *Prod. Plan. Control.* **2023**, 1–15. [\[CrossRef\]](#)
59. Alshahrani, H.; Islam, N.; Syed, D.; Sulaiman, A.; Al Reshan, M.S.; Rajab, K.; Shaikh, A.; Shuja-Uddin, J.; Soomro, A. Sustainability in Blockchain: A Systematic Literature Review on Scalability and Power Consumption Issues. *Energies* **2023**, *16*, 1510. [\[CrossRef\]](#)
60. Kohli, V.; Chakravarty, S.; Chamola, V.; Sangwan, K.S.; Zeadally, S. An analysis of energy consumption and carbon footprints of cryptocurrencies and possible solutions. *Digit. Commun. Netw.* **2023**, *9*, 79–89. [\[CrossRef\]](#)
61. Roy, S.; Rudra, A.; Verma, A. An energy complexity model for algorithms. In Proceedings of the 4th Conference on Innovations in Theoretical Computer Science, New York, NY, USA, 9–12 January 2013; pp. 283–304.
62. Santos, M.; Saraiva, J.; Porkoláb, Z.; Krupp, D. Energy Consumption Measurement of C/C++ Programs Using Clang Tooling. In Proceedings of the 6th Workshop of Software Quality, Analysis, Monitoring, Improvement, and Applications, Belgrade, Serbia, 11–13 September 2017.
63. McNulty, D.; Hennessy, A.; Li, M.; Armstrong, E.; Ryan, K.M. A review of Li-ion batteries for autonomous mobile robots: Perspectives and outlook for the future. *J. Power Sources* **2022**, *545*, 231943. [\[CrossRef\]](#)

64. Caballero, L.; Perafan, A.; Rinaldy, M.; Percybrooks, W. Predicting the Energy Consumption of a Robot in an Exploration Task Using Optimized Neural Networks. *Electronics* **2021**, *10*, 920. [[CrossRef](#)]
65. Castellon, C.; Roy, S.; Kreidl, P.; Dutta, A.; Bölöni, L. Energy efficient merkle trees for blockchains. In Proceedings of the 2021 IEEE 20th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), Shenyang, China, 20–22 October 2021; pp. 1093–1099.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.