engineering proceedings

Proceeding Paper

The System Architecture and Methods for Efficient Resource-Saving Scheduling in the Fog [†]

Anna Klimenko

Institute of IT and Security Technologies, Kirovogradskaya St. 25-2, 117534 Moscow, Russia; anna_klimenko@mail.ru

† Presented at the 15th International Conference "Intelligent Systems" (INTELS'22), Moscow, Russia, 14–16 December 2022.

Abstract: The problem of resource-saving scheduling in a fog environment is considered in this paper. The objective function of the problem in question presupposes the fog nodes' reliability function maximizing. Therefore, to create a schedule, the following is required: the history of the fog devices' state changes and the search space, which consists of preselected nodes of the cloud-fog broker neighbourhood. The obvious approach to providing the scheduler with this information is to poll the fog nodes, yet this can consume the unacceptable time because of the QoS requirements. In this paper, the system architecture and general methods for efficient resource-saving scheduling is presented. The system is based on distributed ledger element usage, which provides the nodes with the proper awareness about the surroundings. The usage of the distributed ledger allows not only for the creation of the resource-saving schedule but also the reduction of the scheduling problem-solving time, which frees addition time that can be used for the solving of user tasks. The latter also affects the overall resource-saving via reliability. The novelty of this paper consists in the development of the hybrid ledger-based system, which integrates and arranges the elements of various ledger types to solve the newly formulated problem.

Keywords: scheduling problem; optimization; fog computing; distributed ledger

check for updates

Citation: Klimenko, A. The System Architecture and Methods for Efficient Resource-Saving Scheduling in the Fog. *Eng. Proc.* **2023**, *33*, 9. https://doi.org/10.3390/engproc2023033009

Academic Editors: Askhat Diveev, Ivan Zelinka, Arutun Avetisyan and Alexander Ilin

Published: 17 May 2023

1. Introduction

The scheduling problem is known as a problem of high importance in the field of distributed computing, including scheduling in the fog- and edge- computing environments. Resource-saving scheduling is a problem with a particular objective function, that is to maximize the reliability function values of the nodes at the end of the user operation. Moreover, the problem solution must not consume much time due to the quality of service (QoS) and quality of experience (QoE) requirements for the fog or edge systems.

Fog computing presupposes the cloud-fog broker (CFB) functioning in the fog layer of the network. The general tasks of the CFB are to receive the computational problem and data from the edge device; to schedule this problem, assigning the subtasks to the fog devices nearby—or if there are no devices with appropriate amount of resources in the fog,to send the problem and data to the cloud—and to return a result to the user's device [1].

There are some issues in this scheme that created by the peculiarities of the fog layer and of the resource-saving scheduling problem:

- There can be an uncertainty about the fog nodes in the CFB neighbourhood. The main reason for this is the dynamic nature of the fog layer [2,3];
- The distances between fog nodes affect the time of data exchange between the fog nodes. Time consumption leads to the impossibility of meeting the QoS requirements and also worsens the reliability functions of the fog nodes because of possible overloading;

- The scheduling problem itself must be solved as soon as possible because of the need to maintain the QoE and QoS requirements;
- Finally, the forming of a resource-saving schedule requires the nodes' workload history, at least when the log of the device resource states changes.

The issues listed above generate the following tasks which must be performed to generate the schedule:

- The search space for the optimization problem solving must be formed on the basis of available fog nodes;
- The workload history must be used for the schedule formation.

The most obvious solution to implementing the scheduling process under the listed conditions is to request all the needed data from the fog nodes in the CFB neighbourhood, and, if sufficient resources are there, to assign the tasks to the node. Yet, distances between nodes can produce unacceptable delays for the formation of scheduling problem search space (as well as for workload information retrieval), while the increase of the scheduling formation time leads to the decrease of the time for the computational task solution. Therefore, the general goal of this paper is to design the architecture and basic functioning methods of the system for efficient resource-saving scheduling in the fog.

The following tasks are considered to achieve the general goal:

- Resource-saving scheduling problem formalization and analysis;
- A state-of-the-art analysis in the field of scheduling in the fog, resource saving, and distributed ledger;
- A comparison of the approaches to providing CFB with the appropriate data;
- System architecture and basic functional method development.

The main contribution of this paper is the development of the architecture and basic functional methods of the system for efficient resource-saving scheduling in the fog. The novelty of this paper consists in the development of the hybrid ledger-based system, which integrates and arranges the elements of various ledger types to solve the newly formulated problem.

2. The Resource-Saving Scheduling Problem and Its Analysis

The scheduling problem in the fog environment takes place when the CFB distributes the computational tasks within the set of nearby fog nodes [1].

The process of task scheduling in general involves np-hard problem-solving, which can be implemented via various methods, including up-to-date heuristics and methaheuristics (genetic algorithms, ant colony algorithms, simulated annealing, etc.). These methods are quite efficient; however, no discrete optimization can be performed without the formation of a search space [4], and a schedule-forming procedure has its time restrictions: with the increase of scheduling time the time for functional tasks implementation reduces. Moreover, the resource-scheduling problem requires the data on the workload history of the nodes because this is the key parameter for the workload distribution.

Consider the CFB, which receives the user task the and data to solve. The problem is to schedule the task within the formed fog node community, so as $P_0(\tau) \to max,\ where\ P_0(\tau)$ is the overall reliability function value of the fog nodes community, including CFB,τ is the moment of the user operation completion.

Consider the network graph $G = <V, U>$ where V is a set of computational nodes, and R is a set of ribs. $V = v_i = <i, p_i, R_i(t_0), L_i>$, where i is the node identifier, p_i is the node performance, $R_i(t_0)$ is the computational resource of the node at the moment of scheduling problem solution, and L_i—is the workload of the node at the moment of t_0. $U = u_i$, where u_i is the data transmission rate of the network rib j. The user operation is described as an acyclic graph, whose vertexes are assigned to tasks and whose ribs are assigned to information connections between them. $O = <T, C>$, where T is the set of subtasks, and C is the set of information connections. $T = t_j = <j, w_j, d_j>$, where j is

the subtask identifier, w_j is the computational complexity of the subtask, and d_j is the data volume transferred to the network. The problem solution is the following tasks assignment: $A = \begin{bmatrix} t_{11} & \dots & t_{1m} \\ \dots & \dots & \dots \\ \dots & t_{ij} & t_{nm} \end{bmatrix}$ such as $P_0(\tau) \to max.$ where $P_0(\tau) = \prod_i P_i(\tau)$, $P_i(\tau) = \exp\left(-\lambda_0 \tau 2^{(kD_j/10)}\right)$.
where $P_j(\tau)$ is the reliability function value,
D is the node workload;
k is the coefficient of node temperature increase depending on the current workload, and
t_{i_j} is the moment of assignment of task j to the node i.

The constraint for this problem is as follows: $\tau < t_c onst$; that is, the user operation completion time must be less than the declared time for this operation. One can see that the formal presentation of the resource-saving scheduling problem is quite common for the scheduling ones and can be solved by one of the existing and described methods (e.g., simulated annealing or some greedy approaches).

However, to solve this optimization problem in the fog, the following data must be provided:

- The search space, which is the set of nodes that are available for the task assignment;
- The workload story for the fog nodes, which is needed to estimate the reliability function values, as it is the integral part of the overall objective function.

Therefore, to solve the resource-saving scheduling problem in the fog, some means must be developed to provide the CFBs with the information required.

3. State-of-the-Art Analysis

The basic research area of this paper covers the following problem fields:

- The computational resources, node failure rate and reliability function connected to the nodes workload;
- The particular scheduling problem-solving under the time constraints and the uncertainty of the resources;
- The distributed ledger technologies and their application to the node information provisioning.

We consider the field of the computational resources, failure rate, and reliability. In the current paper the term "computational resource" refers to the reliability function of the computational node. The reliability function value depends on the failure rate, while the failure rate is related to the device temperature and workload:

$$\lambda = \lambda_0 * 2^{(\Delta T/10)}; \tag{1}$$

where λ is a resulting failure rate, λ_0 is the failure rate under conditions of the unloaded device, and ΔT is the temperature difference between the temperature of the unloaded device and the temperature of the loaded one.

In the study [5], the coefficient is determined, which connects the node temperature and the workload. Consequently, the reliability function is determined as follows:

$$P_i(\tau) = \exp(-\lambda_0 \tau 2^{(kD/10)}). \tag{2}$$

Therefore, by varying the node workload, the reliability function values can be improved at the particular time moment. This approach to system reliability improvement has been used in other studies [6,7], with a variation of objective function forms. However, no attention has been paid to the workload distribution under the uncertainty of the search space, which is highly related to the fog environment.

The scheduling problem is considered carefully in a wide range of publications because the problem itself is not new [8]. Additionally, some studies have examined resource planning and scheduling in the fog but have not considered the questions of reliability, resource-saving, or scheduling under the search space uncertainty.

Notably, in [9], the method of the online formation of the search space in the dynamic fog environment with the usage of onthologies to speed up the search space formation is considered. However, this approach presupposes the formation of the search space online by means of node polling, which may be unacceptable in conditions with time and the QoS restrictions.

Distributed ledger technologies, which first emerged as cryptocurrency systems, have been applied to the particular areas of fog and edge computing, with some examples listed below:

- Security issues in the fog [10];
- Data management [11];
- Consensus mechanisms [12].

Regarding the current paper, distributed ledger technologies can provide synchronized local data storing as the source of available resources, node states, and workload history data. It must be mentioned that as there is no report of ledger usage aimed at improving node reliability,except for in [13]. In this study, the distributed ledger is used to store the information about the device's workload, yet there are no methods providing the data integrity and system functioning in general.

Therefore, despite the rare efforts, there are still no basic approaches to the design and development of particular systems for efficient problem-solving of resource-saving scheduling in the fog.

4. A Comparison of the Approaches to CFB Data Provisioning and the Approach Choice for System Design

As mentioned in the previous section, to solve the resource-saving scheduling problem the following is required:

- A search space for the optimization of problem-solving;
- Information about the workload history of the fog nodes.

All these requirements can be met by means of particular data storing on the fog nodes.

Estimate the time t_0 as the time when all the needed data will be gathered from the nearby fog bodes. The minimum possible value of t_0 is the minimum of t_i, which is the time of information gathering from the nearest fog node.Thus, the lower estimation is as follows: $t_0 \geq t_{\min}$. The estimations of the time needed for the information gathering are as follows, with D_k as the distance between CFB and the fog device:

$$t_{\min} \leq t_0 \leq \sum_{i=1}^{n} t_{\min} D_k \tag{3}$$

Consider a case in which all the needed information is stored on the fog node locally. There is no polling, and the time of information gathering for the search space forming and for the node workload history includes the time of local storage analysis.
Therefore, the estimation of the time for the local data processing and analysis can be as follows:

$$t_0 = \xi(V_{ledger}), \tag{4}$$

where V_{ledger} is the volume of the local data storage, and $\xi(V_{ledger})$ is the time of ledger analysis. Thus, the polling strategy implementation depends on the network diameter and nodes as well as the data transmission channels state, while the local storage of the appropriate information makes it possible to form the search space and analyse the workload history of the nodes much more efficiently. The comparison between the approaches to the information gathering is shown in Figure 1

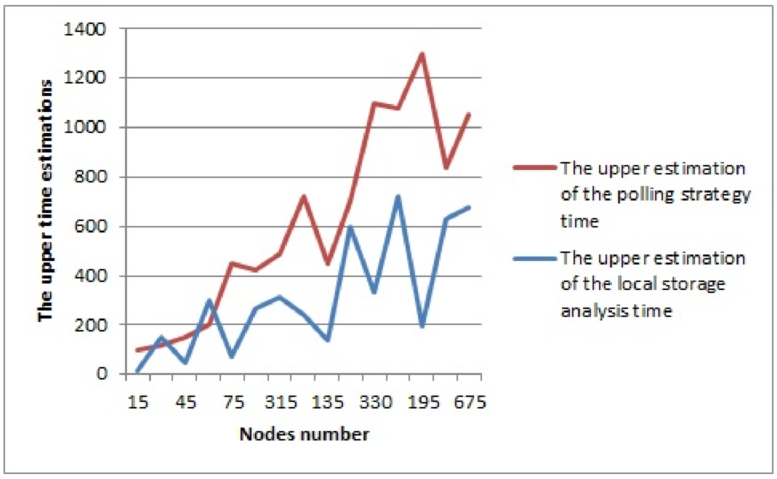

Figure 1. The time estimations for the polling strategy and the local data storage.

The following must be highlighted: the time decrease of the resource-saving scheduling problem-solving increases the resources of the fog nodes due to the possibility of decreasing the workload.

It is well-known that there is a wide range of distributed ledger types, each which use different data storage and consensus methods. For example, there are many consensus methods for blockchain-based ledgers, including proof of work, proof of stake, proof of authority (the competitive consensus), and PBFT (cooperative). Some consensus methods have been developed for ledgers with other storage types, e.g., Nano (block lattice), Swirlds Hashgraph consensus algorithm (Hedera Hashgraph), and others [14].

To provide the CFBs with the appropriate data, the choice of data storage method and the method of consensus on data must be formed on the basis of the general functional requirements of the resource-saving scheduling problem-solving system architecture:

- There are three node types in the system: user (edge) nodes, fog nodes, and CFB nodes. User nodes are the sources of the tasks and data,fog nodes are supposed to participate in the tasks and data processing, and CFB nodes produce schedules and distribute computational workload through the fog nodes and cloud.
- The need to deal with the local copies of the workload history and resource states forms the main requirement to the distributed ledger functioning: the data on the resource state change of the fog node must be placed into the ledger in an order of events.
- The additional requirement for the fog node information collecting and disseminating is that information about the node resource state change must be disseminated through the network as soon as possible.
- In the case of assigning the fog nodes to the CFB, there is a need to detect CFB failure and to restore the information provisioning as soon as possible.

block lattice/Nano technology of data storing and synchronization is the most appropriate for the storing of the changing resource and workload states of the fog nodes. Yet, it is insufficient to synchronize the data only: there must be a mechanism to detect the CFB failure and to recover the system of fog brokers. The view stamped replication concept has potential in this regard.As we have the fully replicated data, there is no need for an exchange with the leader elections procedure. Thus, the time needed for leader election is acceptable.

5. Development of the System Architecture and Basic Functional Methods

Consider the atomic transaction as follows: $Q = < timestamp, Load, node_{id}, t, broker_{id} >$, where $transaction_{id}$ is the transaction identifier, $round_{id}$ is the epoch number, $Load$ is the workload estimation of the node, $Node_{id}$ is the unique fog node identifier, t is the moment the device exploitation begins, and $broker_{id}$ is fog broker identifier. Transactions of the described structure are stored in the block lattice—type ledger. Each fog node is assigned to its own blockchain within the block lattice. Each blockchain implements the storage of the nodes' states. Every time the node changes its state (e.g., workload change), this fact is placed into its assigned blockchain and disseminates through the fog broker nodes, which are the owners of the block lattice storage.

The basic architecture of the system is presented in the Figure 2.

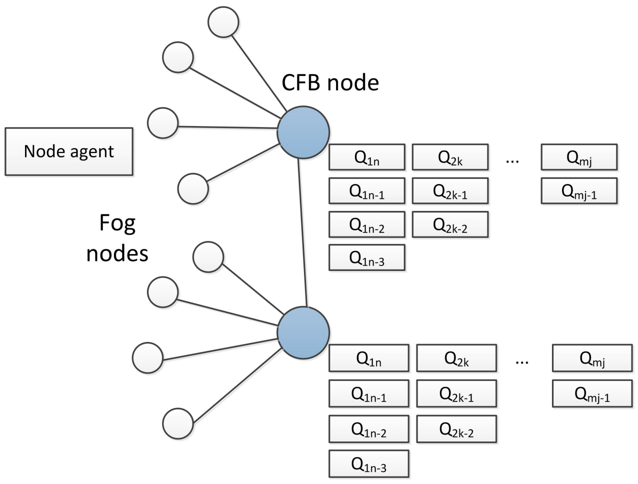

Figure 2. The basic architecture of the system.

The general system architecture implements the following functionality. The fog device software performs the following:

- Provides the node state registration;
- Provides the storage of the device-broker assignment information;
- Provides the fog node–broker interaction;
- Provides the procedures of receiving and processing the computational tasks as planned by a fog broker.

The CFB software performs the following:

- Interaction with other brokers;
- Interaction with the fog nodes;
- Interaction with the ledger replicas and the providing of its functioning;
- Scheduling problem-solving.

Figure 3 contains the software architecture of the fog and broker node.

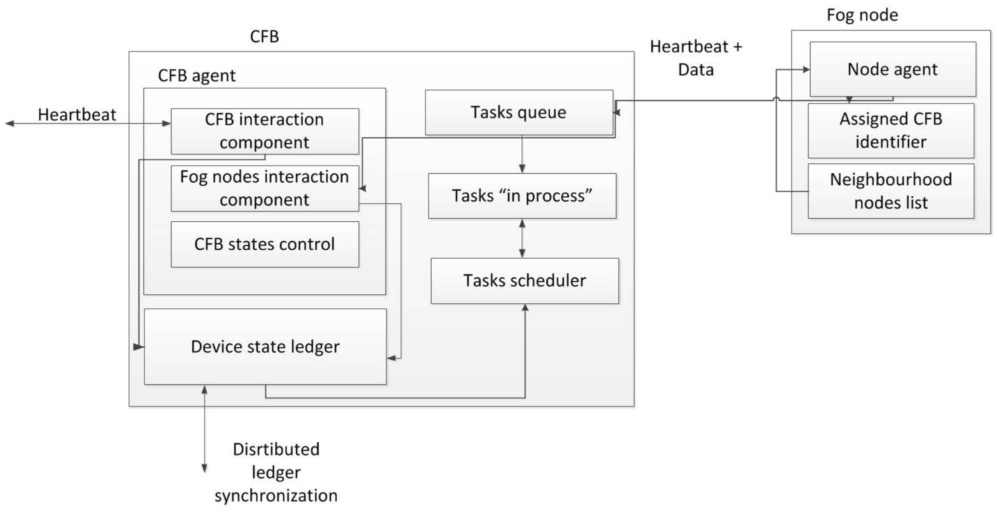

Figure 3. The structure of the CFB component and its interaction with the fog device.

The basic functions of the system are implemented by means of the following methods.

5.1. Adding a New Broker Node to the Broker's Network

1. The broker agent initializes.
2. Broker agent sends a request to the neighbour nodes about their participation in the process of information delivery for the resource-saving scheduling problem-solving. If some nodes are found, then the following occurs:
    - Request of the list of fog brokers from the neighbour fog node.
    - Sending of a request of the ledger copy to the nearest fog broker node.
    - Addition of the new own blockchain to the block lattice, which is assigned to the new device in the fog layer.
3. If there is no devices with ledger copies, then, the current fog broker is the first ledger copy.
4. Creation the first ledger copy with the blockchain assigned to the broker node.
5. Creation of the new list of brokers, each with its identifier being added to the list.
6. Implementation of the search through the network cluster to gather the information about the fog nodes' current states.
7. Addition of new blockchains to the ledger and their being assigned with the new-found nodes.
8. Sending of the broker ID to the active fog nodes.

5.2. Broker Failure

The failure of the fog broker does not make it necessary to remove its own blockchain from the ledger. However, the broker failure can become the cause errors in the scheduling problem-solving when the neighbourhood fog nodes send their state descriptions to the failed broker. In the case of broker failure, the fog nodes do not have the command to send their data somewhere else. This problem can be solved in various ways:

- The fog node sends the state data to some of the nearest fog brokers. In the case of broker failure, no data are lost, yet the question of data duplication emerges.
- The fog node sends state data to the nearest fog broker.

In the case of broker failure, some state information will be lost, yet this situation can be detected within the fog broker network, and the interaction between the fog nodes and brokers can be recovered. To detect the broker failures, the broker community is formed. The brokers' awareness about their neighbourhood is implemented by means of message exchange. In case of distributed leader usage (viewstamped replication concept), broker nodes send the "heartbeat" messages to the leader and receive the same messages from it. One can see that in the distributed leader approach, the information exchange within the broker network will be quite acceptable.

The main stages of the system functioning based on the distributed leader approach are outlined below.

5.3. Functioning Stage

1. The leader sends "heartbeat" messages to its followers.
2. The follower sends the proof of the message's receipt.

5.4. Follower Failure

1. If the current leader has not received the heartbeat message, then;
2. The leader searches its ledger copy for the fog nodes assigned to the failed broker;
3. The leader searches the fog broker, which is nearest to the fog nodes assigned to the failed broker;
4. The request of the new fog broker setup is sent to the fog nodes;
5. The new assigned broker sends the request for fog node states to the fog nodes;
6. States are put into the ledger copy;

7. The state of the failed fog broker is put into the ledger as a state with the full utilization (no resources available);
8. New data are replicated through all the brokers.

5.5. Leader Failure

1. Followers wait for the "heartbeat" from the leader node;
2. If there is no "heartbeat", the new leader is elected, for instance, by means of some simple procedure (round robin);
3. The search of the new broker, which is near the nodes of the failed broker, is conducted;
4. The request of the new fog broker setup is sent to the fog nodes;
5. The new fog broker sends the state request to the new fog nodes;
6. The received states are put into the ledger copy;
7. The state of the failed broker is put to the ledger as the state of full utilization (the lack of available resources);
8. New data are replicated to the ledger copies.

5.6. New Fog Device Addition

1. Within the new fog node emergence, the fog node agent sends its identifier into the network.
2. The first broker receives this message, sends the confirmation to the new node, and blocks it for the other brokers. From this moment the fog node is presupposed to be assigned to the current broker.
3. The assigned broker requests the node state information and puts it into the ledger.

5.7. Fog Node Failure

In the case of fog node failure, there can be a situation in which the assigned tasks can not be performed. In this case, the fog broker can reschedule the problem, taking into account that fact that the time for user computations has decreased. The failed fog node state is put into the ledger as "no resource available".

6. Conclusions

In the current paper, the architecture and some functional methods are proposed for decreasing the resource-saving scheduling problem time.

The distributed ledger technology, block lattice + Nano concept elements were chosen for the data provisioning implementation, while viewstamped replication protocol elements were chosen for the control and recovery of the CFB network. The architecture and methods proposed provide the CFB nodes with the data needed for the following:

- Search space formation;
- Resource-saving scheduling problem-solving.

Furthermore, the possibility to process the local data without the node polling allows the scheduling problem-solving time to be shortened, thus increasing the time for functional user task solution. This leads to a decrease in fog node's workload decrease, and in this way, the provision of additional time is achieved.

Funding: This study received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The author declares no conflict of interest.

References

1. Xu, J.; Hao, X.; Zhang, R.; Sun, X. A Method Based on the Combination of Laxity and Ant Colony System for Cloud-Fog Task Scheduling. *IEEE Access* **2019**, *7*, 116218–116226. [CrossRef]
2. Alenizi, F.; Rana, O. Fog Computing: Towards Dynamically Controlling the Offloading Threshold and Managing Fog Resources in Online Dynamic Systems. *Preprints.org* **2021**. [CrossRef]
3. Giang, N.; Lea, R. Developing applications in large scale, dynamic fog computing: A case study. *Softw. Pract. Exp.* **2020**, *50*, 519–532. [CrossRef]
4. Zhiglyavsky, A.; Zhilinskas, A. *A Global Extremum Search Methods*; Nauka: Moscow, Russia, 1991; p. 247.
5. Klimenko, A.; Melnik, E. Information and Control Systems with Distributed Ledger Usage: A Reliability Issue. In *Artificial Intelligence in Intelligent Systems: Proceedings of 10th Computer Science On-Line Conference 2021*; Springer International Publishing: New York, NY, USA, 2021. [CrossRef]
6. Klimenko, A.; Kalyaev, I. A Technique to Provide an Efficient System Recovery in the Fog- and Edge-Environments of Robotic Systems. In Proceedings of the Interactive Collaborative Robotics: 6th International Conference, St. Petersburg, Russia, 27–30 September 2021. [CrossRef]
7. Korovin, I.; Melnik, E.; Klimenko, A. The Fog-Computing Based Reliability Enhancement in the Robot Swarm. In Proceedings of the Interactive Collaborative Robotics: 4th International Conference, ICR 2019, Istanbul, Turkey, 20–25 August 2019. [CrossRef]
8. Blazewicz, J.; Moseley, B.; Pesch, E.; Trystram, D.; Zhang, G. New Perspectives in Scheduling Theory. *J. Sched.* **2021**, *24*, 161–169. [CrossRef]
9. Klimenko, A.; Safronenkova, I. An Ontology-Based Approach to the Workload Distribution Problem Solving in Fog-Computing Environment. In *Artificial Intelligence Methods in Intelligent Algorithms: Proceedings of 8th Computer Science On-line Conference 2019*; Springer International Publishing: New York, NY, USA, 2019. [CrossRef]
10. Wu, D.; Ansari, N. A Cooperative Computing Strategy for Blockchain-Secured Fog Computing. *IEEE Internet Things J.* **2020**, *7*, 6603–6609. [CrossRef]
11. Alam, T. IoT-Fog-Blockchain Framework: Opportunities and Challenges. *Int. Fog Comput.* **2020**, *3*, 1–20.[CrossRef]
12. Son, B.; Lee, J.; Jang, H. A Scalable IoT Protocol via an Efficient DAG-Based Distributed Ledger Consensus. *Sustainability* **2020**, *12*, 1529. [CrossRef]
13. Kalyaev, I.; Melnik, E.; Klimenko, A. Distributed Ledger Based Workload Logging in the Robot Swarm. In Proceedings of the Interactive Collaborative Robotics: 4th International Conference, ICR 2019, Istanbul, Turkey, 20–25 August 2019. [CrossRef]
14. Masood, F.; Faridi, A. An Overview of Distributed Ledger Technology and its Applications. *Int. J. Comput. Sci. Eng.* **2018**, *6*. 422–427. [CrossRef]