Blockchain based Task Offloading in Drone-aided Mobile Edge Computing

Shuyun Luo⋆, Hang Li⋆, Zhenyu Wen†ϒ, Bin Qian†, Graham Morgan†, Antonella Longo‡, Omer Rana& and Rajiv Ranjan†
⋆ School of Information Science & Technology, Zhejiang Sci-Tech University, Hangzhou, 310018, China
† School of Computing, Newcastle University, Newcastle upon Tyne NE1 7RU, U.K.
& School of Computer Science and Informatics, Cardiff University, Cardiff, CF24 3AA
‡ University of Salento ϒ Zhenyu Wen is the corresponding author

Abstract—An increasing number of cloud providers now offer Mobile Edge Computing (MEC) services for their customers to support task offloading. This is undertaken to reduce latency associated with forwarding data from IoT devices owned by customers to cloud platforms. However, two challenges remain in existing MEC scenarios: (i) the coverage of MEC services is limited; (ii) there is limited ability to develop an audit trail about which MEC service providers have processed a user's data. A new architecture for automatically offloading user tasks in MEC scenarios is proposed which addresses the two challenges above. The architecture makes use of drones to dynamically cache data generated from IoT devices and forwards this data to MEC servers that participate in a private blockchain network. We use simulation to demonstrate the use of the proposed architecture in practical MEC scenarios, demonstrating both the efficiency of the task offloading process and greater visibility of MEC service providers involved in processing user data.

Index Terms—Mobile Edge Computing, Drone, Blockchain, Task Offloading

1 Introduction

With the development of various mission-critical applications of Internet of Things (IoT), e.g. vehicular networks (both vehicle-to-vehicle and vehicle-to-infrastructure), augmented reality and city sensing, there is a need to ensure sufficient computational capacity and low latency connectivity is available for devices that are used in such applications. However, there is a tension between having sufficient computing resource and low latency in IoT applications. A cloud platform can have sufficient computing resources, however transferring data from IoT devices to cloud-based systems may introduce significant delay. This is particularly true in closer proximity to IoT devices, where the first hop network from the IoT device may have limited network capacity. A Mobile Edge Computing (MEC) framework is proposed to address some the above issues by offloading computation from IoT devices to local/ regional MEC servers instead of using a remote cloud system. This overcomes a number of potential constraints in current systems: lower energy consumption at the terminal devices [1], reduced requirement to transfer security-sensitive data to a cloud platform, and the need for network capacity between the IoT device and the cloud platform (over a multi-hop connection).

Many existing offloading strategies for MEC environments focus on maximizing applications' performance by partitioning computational tasks across IoT devices, MEC servers, and a cloud platform. However, there is limited coverage on development of the MEC network itself – which is often assumed to be made of homogeneous types of devices/ resources. In rural environments and emergency relief scenarios, for example, the number of MEC servers are very limited. As a result, not all IoT devices can be covered by the MEC network. Inspired by [2]–[4] that use drones (Unmanned Aerial Vehicles (UAVs)) to cache data generated from IoT devices that cannot be reached by cellular networks, we also describe the use of drones to forward cached data from IoT devices to MEC servers, rather than a cloud platform. The aim is to reduce the latency of data transmission via cellular networks.

In addition, MEC server may be owned and operated by various organizations (e.g. Huawei, Google, Microsoft) and these providers need to work collaboratively to offer the "best" service to their costumers. This also raises a significant challenge of how to improve the visibility of the MEC service providers for handling users' sensitive data, or ensure some standard security and privacy requirements such as General Data Protection Regulation (GDPR) are met. To achieve this, we use a blockchain network to improve the visibility of MEC service providers, helping increase trust from their customers. Moreover, we choose a permissioned, private blockchain to meet two essential requirements of our system: (i) offloading of tasks to the "best" MEC server; (ii) developing a non-modifiable audit trail of which MEC server has been involved in processing user data (identifying ownership and processing carried by the MEC server).

In this paper, we present a new secure offloading system that utilizes drones to extend the coverage of MEC networks and a private Blockchain is used to ensure the visibility and accountability of the MEC server providers on operating users' data while guaranteeing the performance of each offloading task. The proposed system has the following advantages:

- **Cost efficiency & accessibility**: Using drones to cache data from IoT devices, and forwarding this

IEEE Network

to a MEC network that cannot be directly reached from the IoT device. This saves cost to deploy MEC servers, while ensuring coverage of MEC services.

- **Trustworthiness**: The private blockchain only allows a certain members to participate in a pre-identified, permissioned network and the participants must follow restrictions or policies in the network. This can filter the untrustworthy MEC service provider.
- **Visibility**: Blockchain is a decentralized system that secure data based on its completely transparent and verifiable property. Thus, in our proposed system, all the operations performed to users' data are recorded and can be verified, including which drone cached the data, which MEC server processed the data and what kind of analytic tasks are performed etc.

To the best of our knowledge, this is the first work to adopt the blockchain technology in the drone-aided MEC scenarios. The main intellectual contributions of this work are summarized as follows.

1) A new decentralized offloading architecture in Drone-aided MEC(DMEC) framework is designed to improve the coverage of MEC services.
2) A new smart contract is designed to integrate with offloading policy for DMEC framework to ensure the visibility of MEC service providers involved in processing customers' data.
3) Simulation results demonstrate the feasibility of the proposed architecture in practical DMEC scenarios.

The remainder of this article is organized as follows. We review related work in §2. The system architecture is described in §3. Next, we propose a new smart contract to interact with offloading policy in MEC scenarios in §4. In §6, experimental results show the feasibility of our architecture. We present a security analysis of our architecture in §5. Finally, we conclude this paper and look forward the future work.

2 RELATED WORK

The contribution of this work lies in the collaborative combination of three important cutting-edge technologies, that is offloading in MEC, UAV and blockchain. This section explores the previous research carried out by combining two of those technologies.

2.1 Offloading in MEC and Blockchain

Amounts of existing research [5]–[7] focus on offloading the blockchain mining tasks from IoT devices to the MEC servers. Chen. et al [6] extended the offloading process to multi-hop network. Jiang. et al [7] considered two scenarios with both fixed and dynamic number of miners by formulating a multi-leader multi-follower Stackelberg game. [8] exploited using blockchain to ensure data integrity, while ignoring other security threats. [9] designed two smart contracts to trade the computing resource and loan coin for mobile equipment. [10] presented a blockchain-based MEC framework for adaptive resource allocation and computation offloading where the blockchain is responsible for the management and control functions. However, when above frameworks of block-chain-empowered MEC integrate blockchain technology into IoT devices, they fail to consider the resource-constrained IoT devices for supporting the computation-consuming blockchain mining tasks. Our architecture avoids the mining burden of IoT devices and increases the flexibility for applications in a large scale of IoT device scenarios.

2.2 Offloading in MEC and UAV

UAVs feature broader communication coverage and are thus considered as relaying services providing computation offloading for mobile users in MEC scenarios [4]. Also, the UAV's trajectory can be optimized minimizing the overall energy consumption. [11] presented an UAV-aided mobile edge computing (UMEC) model, where an UAV with certain computing power is leveraged to relieve the communication and computing burden on the edge clouds. [12] considered computation offloading to both UAV and MEC. The UAV agent perceives and intelligently minimizes task execution latency as well as the energy consumption. UAVs are highly flexible, operable and response-sensitive. The above research works took advantages of these features, but failed to address the security challenges of UAVs. We benefit from the smart contract and give specific design for tackling the potential threats in UAVs.

2.3 UAV and Blockchain

[13] addressed the poisoned content discovery problems in Named Data Networking (NDN) using UAVs. They integrated the interest-key-content binding, forwarding policy and on-demand verification together to discover poisoned content. In order to reduce the high overheads in hierarchical networks, [14] used UAVs as on-demand nodes and presented a novel drone-caching framework to ensure ultra-reliable communications. The above methods are not very practical, which requires the cooperation of three blockchain. In our design, the UAV is apart from blockchain network, only playing the role of offloading hubs. The blockchain network is used to improve the users' trust for MEC service providers.

3 SYSTEM ARCHITECTURE

In this section, we describe the architecture of the decentralized offloading system in detail. The architecture is composed of three layers with the flow of the data: *Wireless Sensor Networks (WSNs)* layer, *Drones* layer and *MEC Servers* layer. *WSNs* are data generators usually deployed across various environments such as buildings, streets and forests. The collected data is used for applications like smart building, intelligent transportation and fire alarm system etc. *Drones* layer acts as the offloading hub for catching and forwarding the data from the WSNs to the MEC servers where data processing task proceeds. The processing tasks are offloaded to the "best" MEC server with the minimum task latency. Finally, in *MEC Servers* layer, a closed blockchain is set up among MEC servers for auditing service provider's honest during user data operation.

Figure 1 illustrates the architecture of the decentralized offloading system with an example to explain the implementation details. As shown by the orange line, WSN_2

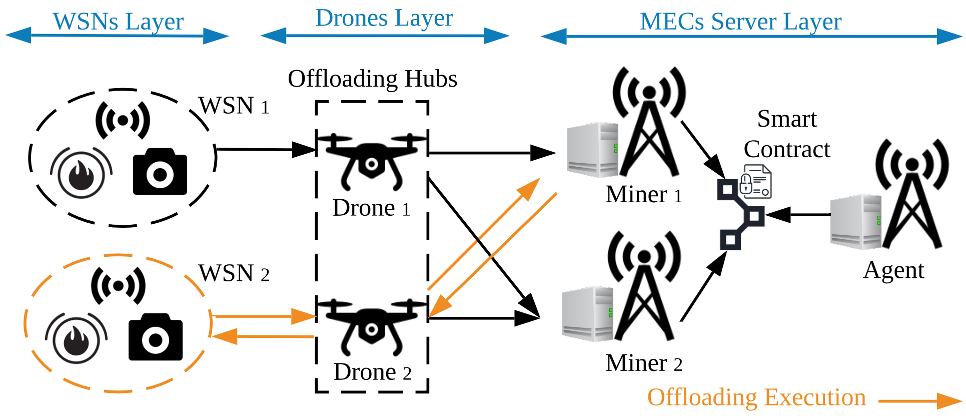

Fig. 1: Decentralized Offloading System

generates a task and forward it to $Drone_2$. Next in the $Drone_2$ offloading hub, the smart contract decides that the task is offloaded to $Miner_1$ for analysis and proceeds accordingly. After the computation is finished, the results is sent all the way back to the original location where the data is generated. We explain the main components for each layer that jointly execute the aforementioned example:

- Wireless Sensor Networks (WSNs) layer: This layer usually contains multiple IoT sensors collecting data from physical environment for various IoT applications. The IoT devices within WSNs usually have limited computational power, memory and energy storage which necessitates the offloading operation.
- Offloading Hubs (Drones layer): In our architecture, a drone is used as an offloading hub and is responsible for: 1) relaying the offloading tasks to an appropriate MEC server, 2) translating the inter WSNs-blockchain communication protocol. Specifically, all drone and MEC servers are interconnected, with each drone receiving data from multiple WSNs (IoT devices) IoT devices will only be able to request offloading information from the blockchain using the offloading hub.
- Blockchain network (*MEC Servers* layer): In the *MEC Servers* layer, a blockchain network is deployed with a smart contract committing data offloading policy. The blockchain network contains two types of nodes: the *Agent* and the *Miner*. The *Agent* is responsible for generating and deploying the smart contract to the blockchain network, such that each node in the blockchain can execute the smart contract automatically. The offloading tasks are then assigned to smart contract qualified *Miner* nodes. All operations from the assigned *Miner* are recorded to blockchain for further verification. The blockchain network in our architecture is designed as a private blockchain for its specific functionality. We chose a private blockchain since it has extra access limitation for participators to guarantee the authority of MEC servers.

4 System Design

This section gives a depiction of the operations designed in the smart contract and the interaction procedure between the components of our architecture. Moreover, we give some discussion about the system limitations.

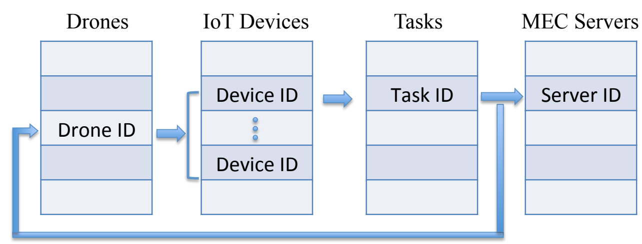

Fig. 2: Data Structure in the Smart Contract

4.1 Smart Contract

Figure 2 shows the data structure in the smart contract. Each drone covers one or multiple WSNs and takes responsibility for a set of IoT devices in the WSNs, and each IoT device generates at most one offloading task at one time. The task will be offloaded to a specific MEC server by smart contract through a drone.

It is assumed that I is the set of public keys of $I(d)$ of each drone d, G is the set of the public keys $G(m)$ of each IoT device m and P is the set of offloading policies where p refers to the specific offloading rules to select the "best" MEC server to offload tasks from IoT devices.

In the smart contract, the drones and IoT devices are identified in the offloading system by their public keys. The order of components registration is depicted below. First MEC servers are signed in by the operation $RegisterServer(Q(s))$, next the drones are joined by $RegisterDrone(Q(s), I(d))$ iff drone d is authorized by MEC server s. After that, the IoT devices are registered by $RegisterDevice(I(d), G(m))$ iff drone d is the offloading hub of device m. Besides, the tasks will be identified by their IDs. The mapping from the drone to the IoT device is done by $AddDronetoDevice(I(d), G(m), Q(s), p)$ iff drone d is the offloading hub of device m. The offloading policy can be deployed into the smart contract by $AddOffloadingPolicy(I(d), G(m), Q(s), p)$, which determines how to choose the MEC server to offload. When it needs to find the offloading hub of the specific IoT device, $QueryDrone(I(d))$ can be called.

$QueryOffloading(I(d), t, G(m), Q(s), p)$ is used to obtain the result of offloading task t by executing offloading policy p. It is worthy to mention that the fetching process does not incur any fee or latency, since drones obtain the information from the blockchain store directly, rather than use a transaction.

4.2 Offloading Policy

The offloading policy plays a crucial part as it determines which local MEC server would be performed the offloading computation task. Currently, the design of offloading policies usually takes multiple factors into consideration together, such as the distance of the request device and the MEC server, the access and computation capacity of MEC servers, the security level of MEC servers, and the availability of wireless connection links, etc. In this paper, we only consider some simple offloading policies as discussed in the following.

4.2.1 The Random Policy

When the drone has no prior knowledge about the MEC servers, the drone will choose randomly one MEC server to offload the task delivered from IoT devices.

4.2.2 The Nearest Policy

The drone will connect with the nearest available MEC server in the blockchain network, a miner node in Figure 1. The nearest policy indicates that drones always find the nearest MEC server to offload the task from IoT devices.

4.2.3 The Max Computing Capacity (MCC) Policy

In max computing capacity policy, the drones always choose the MEC server with the maximum computing capacity to offload tasks.

4.2.4 Smart Contract Policy

In our smart contract policy, it aims to find the best offloading MEC server with minimum latency. Since the scale of the blockchain network is so small, we assume that the consensus time is negligible or with little difference among different policies. Thus, we focus on the task transmission delay and computation offloading delay. Given that a_t is the data amount of offloading task t, $\bar{r}$ the average transmission rate, b_t is the computation amount of offloading task and $\bar{c}$ the average computing capacity, the drone will choose the MEC server by the Nearest policy if $a_t/\bar{r} > b_t/\bar{c}$, otherwise by the MCC policy.

4.3 System Interactions

Figure 3 illustrates the interactions among the system components, which can be divided into three phases: establishing the blockchain network, signing the drones and IoT devices into the system, and executing the offloading operation for the tasks from IoT devices.

4.3.1 Network Establishment

The blockchain network is established among the network of MEC servers in the beginning. Once the blockchain network is created, the agent node takes the responsibility to deploy the smart contract into the blockchain network. The smart contract defines all the operations of the offloading policy and it will generate an unique address to identify this smart contract when it is accepted by the blockchain network. All components in the offloading system use that unique address to interact with the smart contract and execute the operations automatically designed in the smart contract. For example, all the drones in the system need to register as offloading hubs by interacting with the smart contract.

Figure 3 shows how a drone interact with the address and query the Agent node. The offloading hub connects with several available MEC servers in the blockchain network, i.e., miner nodes. Each miner hosts a distributed copy of the blockchain to make all the operations to offloaded data accountable.

4.3.2 Registration

Any MEC server in the offloading system can be registered into the blockchain network. Before a drone is registered as a offloading hub, it needs to be authorized by the node of the blockchain network. Then it registers to the blockchain network by interacting with the smart contract. After then, the IoT devices can be registered by the registered drones thought the address of the smart contract. Meanwhile, drones will receive an address of the registered device to identify who is the offloading hub of the device. The $QueryDrone()$ operation can verify the registration of IoT devices under a drone.

4.3.3 Offloading Execution

After the IoT devices and drones are signed into the specific smart contract, the offloading policy is executed automatically for task T_1 of device M_1. The IoT device M_1 first sends the offloading task to its offloading hub, a verified drone, then the drone transfers the data and execution tasks to the selected MEC server for offloading. The MEC server selection is based on the offloading policy in the smart contract. All the processing or operation to users' data as long as the execution results will be publish to the blockchain network. At the same time, the execution results will be sent to the drone and forwarded back to IoT device M_1.

4.4 Tackle of Blockchain's Limitations

In this subsection, we illustrate how our propose system to overcome the limitations of blockchain systems.

4.4.1 Cryptocurrency Fees

In the blockchain platform, the cryptocurrency fees are used to award the miners who successfully manage their mined blocks into the blockchain for all transactions. It is also required a minimum fee for a transaction in some public ledgers to avoid spam transactions. In our case, this fee is covered on users' services fee. All MEC servers are motivated to mine the blockchain.

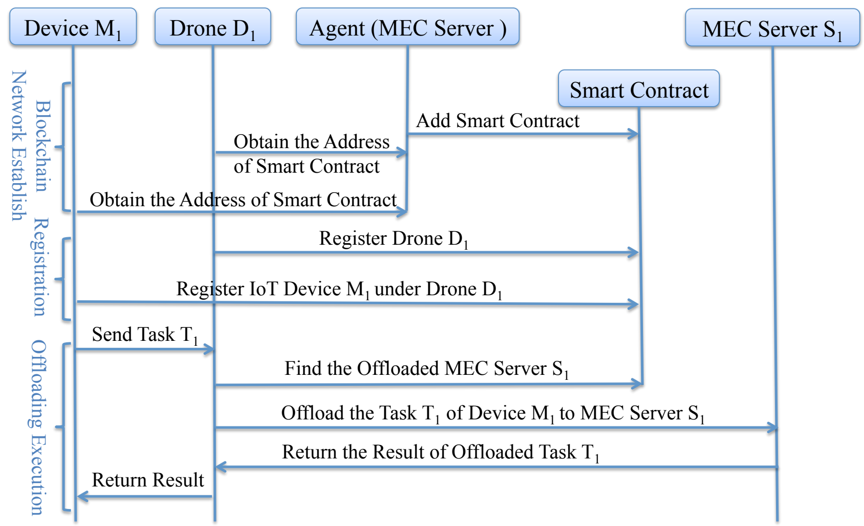

Fig. 3: Network Establishment, Registration and Offloading Execution

4.4.2 Consensus time

It is well known that the consensus procedure in blockchain occupies the most time during a transaction generation. However, in our system the number of MEC server is limited in a private blockchain, thereby significantly reducing the consensus time. If we wait for a offloading task's results writing into blockchain before sending them back to corresponding IoT device, it still introduce huge delay. Therefore, in our design that obtained results can be returned immediately and must be returned by the drone which signs the smart contract. As a result, we solve the delay issue and also prevent the threat that unauthorized drone steal the offloading information.

5 Security Analysis

Security is essential for any management systems, and our offloading system is of no exception. In this section, we identify main threats in our proposed architecture. After that, some solutions are proposed to deal with the threats.

To this end, we use the STRIDE thread model [15] as the security criteria which has six categories: **S**poofing, **T**ampering, **R**epudiation, **I**nformation disclosure, **D**enial of service(DoS), and **E**levation of privilege. Table 1 shows the relations between the threats and the security properties of a system.

Based the model, we list the security threats of our system in Figure 4 and propose some prevention solutions in the following.

Asset	Threats					
	S	T	R	I	D	E
MEC server						
Drone	✓	✓	✓	✓	✓	
IoT Device	✓					

Fig. 4: Potential Security Threats

- MEC Server: Overall, the blockchain guarantees the security of MEC servers, specifically, the data integrity and availability. The authentication process of the blockchain ensures that MEC servers never disclose offloading information. In addition, the traceability property of the blockchian prevents the MEC servers from tampering the results of offloading tasks or repudiating their actions.
- Drone: A malicious drone could spoof (impersonate a normal offloading hub), tamer (change the offloading information during data transmission), repudiate (deny performing an action), DoS (degrade the relay service to IoT devices) or disclose sensitive information of IoT devices. Signature certification could solve the above potential threats. Drones could get signed certificates from a MEC server from the blockchain network, and IoT devices could verify the authentication of the offloading hub by those certificates.
- IoT device: If a trust device is impersonated by a malicious IoT device, it may spread poison content into the offloading system, which consumes the network resources and has a great impact on the performance of task offloading. The authentication of IoT devices can also avoid the above security threats.

6 Performance Evaluation

In this section, we aim to evaluate the feasibility of the proposed architecture which is suitable for any off-loading policy.

Experiment setup. We implemented the contract in Solidity 0.4.26 [1] on the test net of Ethereum network, Ropsten [2], which is a popular blockchain test net for evaluation of smart contracts. Moreover, we emulate a scenario that one drone carries a set of tasks to be offloaded to three MEC servers. Four policies discussed in section 3.2 are developed in our smart contract and are used to generate offloading solutions for the submitted tasks.

1. https://solidity.readthedocs.io/en/v0.4.24/
2. https://ropsten.etherscan.io/

TABLE 1: Threats and Security Properties

Threat	Property	Description
Spoofing	Authentication	The identity of users is established.
Tampering	Integrity	Data can only be modified by verified person.
Repudiation	Non-repudiation	Not denial actions are allowed.
Information disclosure	Confidentiality	Control the access to the data.
Denial of service	Availability	System is ready for receiving tasks.
Elevation of privilege	Authorization	Identify the users who can access the data.

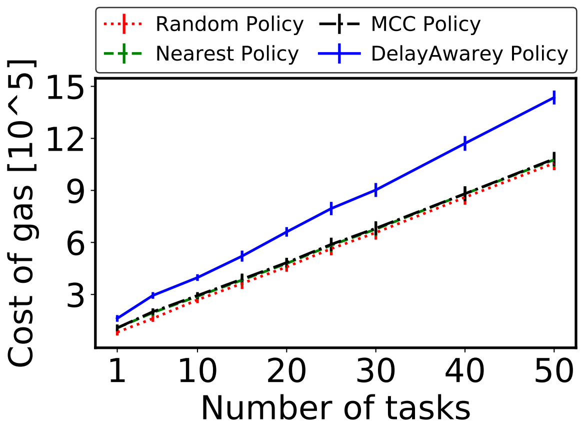

Fig. 5: The Gas Consumption under Different Offloading Policies

In our smart contract, we define an interface $GetTaskInfo()$ to capture the task information, including data amount, computation amount and ID, meanwhile anther interface $GetServerInfo()$ is defined to obtain the MEC server information, such as MEC-Drone distance, computation capacity and address.

Evaluation metrics. We measure the performance of the policies via "Gas" which is the cost for the miners to execute the transactions. This cost depends on the complexity of each policies, i.e., how much computation and storage resources are consumed for running the smart contract. Each experiment is repeated 20 times, and the average values and their variance are reported.

Figure 5 demonstrates the experimental results, where the DelayAware policy consumes more gas than others with the increasing number of tasks. For convenience of analysis, the number of offloading tasks is assumed as n, while the number of MEC servers is represented as m. The reason is that the time complexity of the DealyAware policy is $O(2n + m)$, relatively simpler than other policies, i.e., the random policy is in $O(n)$ time, the Nearest policy and MCC policy are the same with the time complexity of $O(n+m)$. In this experiment, m is a constant, equal to 3. Hence, the gas consumption of all four policies is linearly increasing with the number of offloading tasks.

7 Conclusion and Future Work

In this paper, we present a novel architecture for automatically distributed offloading systems in droned-based MEC scenarios. The architecture brings in the drone as the offloading hub to detect the authorized IoT devices and help them to offload tasks to MEC servers based on the blockchain technology. The architecture supports the mobility of IoT devices and allow them to join or leave at any time. To evaluate the system performance, we focus on the offloading latency introduced by the smart contract, and the results demonstrate that the high reliable offloading policy deployed on the smart contract is feasible in specific scalable MEC scenarios.

Our future work focuses on two parts:

1) Establish a real private blockchain network to evaluate the practical performances, such as the latency and overhead of smart contracts.
2) Design more intelligent offloading policy to apply for more complex scenarios, such as the drones have high mobility to collect the data from IoT devices.

References

[1] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1628–1656, 2017.

[2] M. Chen, M. Mozaffari, W. Saad, C. Yin, M. Debbah, and C. S. Hong, "Caching in the sky: Proactive deployment of cache-enabled unmanned aerial vehicles for optimized quality-of-experience," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 5, pp. 1046–1061, 2017.

[3] Y. Zeng, R. Zhang, and T. J. Lim, "Wireless communications with unmanned aerial vehicles: Opportunities and challenges," *IEEE Communications Magazine*, vol. 54, no. 5, pp. 36–42, 2016.

[4] S. Jeong, O. Simeone, and J. Kang, "Mobile edge computing via a uav-mounted cloudlet: Optimization of bit allocation and path planning," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 3, pp. 2049–2063, 2017.

[5] X. Qiu, L. Liu, W. Chen, Z. Hong, and Z. Zheng, "Online deep reinforcement learning for computation offloading in blockchain-empowered mobile edge computing," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 8, pp. 8050–8062, 2019.

[6] W. Chen, Z. Zhang, Z. Hong, C. Chen, J. Wu, S. Maharjan, Z. Zheng, and Y. Zhang, "Cooperative and distributed computation offloading for blockchain-empowered industrial internet of things," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8433–8446, 2019.

[7] S. Jiang, X. Li, and J. Wu, "Hierarchical edge-cloud computing for mobile blockchain mining game," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2019, pp. 1327–1336.

[8] X. Xu, X. Zhang, H. Gao, Y. Xue, L. Qi, and W. Dou, "Become: Blockchain-enabled computation offloading for iot in mobile edge computing," *IEEE Transactions on Industrial Informatics*, 2019.

[9] Z. Zhang, Z. Hong, W. Chen, Z. Zheng, and X. Chen, "Joint computation offloading and coin loaning for blockchain-empowered mobile-edge computing," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 9934–9950, 2019.

[10] F. Guo, F. R. Yu, H. Zhang, H. Ji, M. Liu, and V. C. Leung, "Adaptive resource allocation in future wireless networks with blockchain and mobile edge computing," *IEEE Transactions on Wireless Communications*, 2019.

[11] F. Zhou, Y. Wu, H. Sun, and Z. Chu, "Uav-enabled mobile edge computing: Offloading optimization and trajectory design," in *2018 IEEE International Conference on Communications (ICC)*. IEEE, 2018, pp. 1–6.

[12] R. Wang, Y. Cao, A. Noor, T. A. Alamoudi, and R. Nour, "Agent-enabled task offloading in uav-aided mobile edge computing," *Computer Communications*, vol. 149, pp. 324–331, 2020.
[13] K. Lei, Q. Zhang, J. Lou, B. Bai, and K. Xu, "Securing icn-based uav ad hoc networks with blockchain," *IEEE Communications Magazine*, vol. 57, no. 6, pp. 26–32, 2019.
[14] V. Sharma, I. You, D. N. K. Jayakody, D. G. Reina, and K.-K. R. Choo, "Neural-blockchain-based ultrareliable caching for edge-enabled uav networks," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 10, pp. 5723–5736, 2019.
[15] S. Hernan, S. Lambert, T. Ostwald, and A. Shostack, "Threat modeling-uncover security design flaws using the stride approach," *MSDN Magazine-Louisville*, pp. 68–75, 2006.

Shuyun Luo is a lecture at Zhejiang Sci-Tech University, China. Her research interests include edge computing, reinforcement learning and network economics. Email: shuyunluo@zstu.edu.cn

Hang Li is a master student at Zhejiang Sci-Tech University, China. His research interests include UAV-based edge computing. Email: zstlihang@163.com

Zhenyu Wen is currently a postdoc researcher with the School of Computing, the Newcastle University, UK. His research interests include Multi-objects optimization, Crowdsources, AI and Cloud computing. Email: Zhenyu.Wen@newcastle.ac.uk

Bin Qian is a postgraduate research student in the school of computing, Newcastle University, UK. His research interests include IoT, Machine Learning. Email: b.qian3@ncl.ac.uk

Graham Morgan is a reader in the School of Computing, Newcastle University. Area of expertise: Cloud Computing, Video Games, Distributed Systems, Deep Learning Email: graham.morgan@ncl.ac.uk

Omer Rana is a full professor of performance engineering in the School of Computer Science and Informatics at Cardiff University. His research interests include performance modelling, simulation, IoT, and edge analytics. Contact him at ranaof@cardiff.ac.uk.

Rajiv Ranjan is a Chair and Professor at Newcastle University, UK, and at China University of Geosciences, China. He has expertise in cloud computing, big data, and the Internet of Things. Email: raj.ranjan@ncl.ac.uk