Trusted Routing for Blockchain-Empowered UAV Networks via Multi-Agent Deep Reinforcement Learning

Ziye Jia, Member, IEEE, Sijie He, Qiuming Zhu, Senior Member, IEEE, Wei Wang, Member, IEEE, Qihui Wu, Fellow, IEEE, and Zhu Han, Fellow, IEEE

Abstract—Due to the high flexibility and versatility, unmanned aerial vehicles (UAVs) are leveraged in various fields including surveillance and disaster rescue. However, in UAV networks, routing is vulnerable to malicious damage due to distributed topologies and high dynamics. Hence, ensuring the routing security of UAV networks is challenging. In this paper, we characterize the routing process in a time-varying UAV network with malicious nodes. Specifically, we formulate the routing problem to minimize the total delay, which is an integer linear programming and intractable to solve. Then, to tackle the network security issue, a blockchain-based trust management mechanism (BTMM) is designed to dynamically evaluate trust values and identify lowtrust UAVs. To improve traditional practical Byzantine fault tolerance algorithms in the blockchain, we propose a consensus UAV update mechanism. Besides, considering the local observability, the routing problem is reformulated into a decentralized partially observable Markov decision process. Further, a multi-agent double deep Q-network based routing algorithm is designed to minimize the total delay. Finally, simulations are conducted with attacked UAVs and numerical results show that the delay of the proposed mechanism decreases by 13.39%, 12.74%, and 16.6% than multiagent proximal policy optimal algorithms, multi-agent deep Qnetwork algorithms, and methods without BTMM, respectively.

Index Terms—UAV network, trusted routing, blockchain, multiagent deep reinforcement learning, Byzantine fault tolerance.

I. INTRODUCTION

As a key component of the space-air-ground integrated network in the six generation communication techniques, unmanned aerial vehicles (UAVs) are rapidly developed and widely applied to multiple tasks, such as disaster rescue and real-time monitoring [1], [2]. In these applications, UAVs can serve as aerial base stations for data collection and transmission, providing low-cost, flexible, and versatile services [3]. Besides, UAV networking can cooperatively accomplish complex tasks, in which routing is a significant issue for optimizing the

Ziye Jia is with the College of Electronic and Information Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China, and also with the State Key Laboratory of Integrated Service Networks, Xidian University, Xi'an Shaanxi 710071, China (e-mail: jiaziye@nuaa.edu.cn).

Sijie He, Qiuming Zhu, Wei Wang, and Qihui Wu are with the College of Electronic and Information Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China (e-mail: sx2304109@nuaa.edu.cn; zhuqiuming@nuaa.edu.cn; wei_wang@nuaa.edu.cn; wuqihui@nuaa.edu.cn).

Zhu Han is with the University of Houston, Houston, TX 77004 USA, and also with the Department of Computer Science and Engineering, Kyung Hee University, Seoul 446-701, South Korea (e-mail: hanzhu22@gmail.com).

data transmission [4]–[6]. Moreover, since UAV networks are characterized by the complex application environment, high dynamic and distributed topology, they are vulnerable to malicious attacks [3]. When there exist malicious UAVs, the hostile activities can lead to the degradation of routing performances. Therefore, it is necessary to evaluate the behaviors and manage the malicious UAVs. There exist a couple of works focusing on building the node trust table by a single-node based centralized manager, which is not resilient to fault tolerance and can be easily tampered with [7]. Hence, to improve the routing security, the blockchain technique has been introduced to build decentralized trusted mechanisms [8]. In particular, the blockchain is a distributed ledger, consisting of a chain of blocks, in which transaction details and records are stored and hard to be tempered with [9].

Meanwhile, delay is a key quality of service for routing in the UAV network, since low delay can significantly improve the timeliness and reliability for various emergency applications, such as surveillance information transmission [10]. Besides, the energy and storage capacity of UAV networks are constrained. Hence, it is necessary to collaborate the resources of UAV networks to provide routing services with low delay. Additionally, due to the distributed topology, high mobility, and low-trust of UAV networks, the availability of communication links is susceptible, leading to the reduction of routing performances. Therefore, it is crucial to design a dynamic routing algorithm to maintain reliable and robust communications in UAV networks. Most traditional routing algorithms are designed for static environment or fixed rules, such as the Bellman Ford algorithm and Dijkstra algorithm, which cannot be directly applied to UAV networks [11], [12]. Besides, it is challenging to obtain the global information of the decentralized UAV network [13]. Hence, it is significant to design a dynamic and adaptive algorithm to enable the efficient and safe routing in UAV networks.

To deal with the above challenges, in this paper, due to the dynamic and distributed topology, as well as the complex environment of UAV networks, the routing process is depicted by a time-varying model with malicious UAVs. In light of the constructed UAV network, the routing problem is formulated to minimize the total end-to-end delay, which is an integer linear programming (ILP) and intractable to solve. Besides, to ensure the security of routing among UAVs, a node trust management mechanism based on the blockchain technique is proposed, in which each UAV obtains a trust value with respect to the delivery rate and the correctness of transmission paths. When the value is lower than the given safety threshold, the UAV is isolated from the network. Besides, it is challenging to obtain the global information due to the decentralized UAV network. Hence, we reformulate the routing problem into a decentralized partially observable Markov decision process (Dec-POMDP) and further design a multi-agent double deep Qnetwork (MADDQN)-based approach. Specially, Dec-POMDP is a mathematical framework from the variant of the MDP, for modeling the decentralized multi-agent and partial observability of the environment. In this work, decentralized UAVs act as agents that independently observe local states and jointly optimize routing decisions [14]. Finally, numerical simulations are conducted to verify the performance of the proposed algorithms.

In short, the major contributions of the proposed work are summarized as follows:

- We characterize the routing process via a time-varying UAV network with malicious nodes. Besides, we formulate the routing problem to minimize the total end-to-end delay, which is an ILP and tricky to solve by traditional optimization schemes.
- To deal with the issue of network security, the blockchainbased trust management mechanism (BTMM) is designed, which records the malicious behaviors and trust values of UAVs. Further, to promote the routing security in UAV networks, we design the consensus node update mechanism to improve the practical Byzantine fault tolerance (PBFT) algorithms.
- Since it is challenging to obtain the global information due to the decentralized UAV networks, we reformulate the routing problem into a Dec-POMDP. Furthermore, an MADDQN-based routing algorithm is proposed to learn the dynamically changing network topology and make decisions to minimize the end-to-end routing delay.
- Extensive simulations are conducted to evaluate the proposed BTMM-MADDQN algorithms with attacked UAVs.
 Numerical results show that the proposed algorithm reduces the delay compared to the BTMM-multi-agent proximal policy optimal (BTMM-MAPPO) algorithm, BTMM-multi-agent deep Q-network (BTMM-MADQN) algorithm, and MADDQN algorithm without BTMM when there exist malicious UAVs.

The rest of this paper is organized as follows. In Section II, we introduce the related works. The system model and optimization problem are illustrated in Section III. Then, we design the algorithms in Section IV. Simulations are conducted and the performance analyses are shown in Section V. Finally, the conclusions are drawn in Section VI.

II. RELATED WORKS

There exist a couple of works related to the trust management. For instance, the authors in [15] propose trust attestation

mechanisms based on a centralized authority to evaluate the trust of a network slice. To promote the security and efficiency for edge devices, [16] presents a dynamic trust management model based on the feedback, in which a third-party trust agent is set up to cooperatively manage the trust and a cloud center is assumed to be reliable. Authors in [17] evaluate the trust via a multi-trust method, in which the base station is considered as a central command authority that cannot be compromised by an attacker. In [18], a lightweight trust management is presented to overcome the security issue, which utilizes a centralized trust agent to maintain a database that stores the trust degree. As above, the centralized trust management approaches are not applicable to distributed UAV networks. Although these works can enhance the network security by managing the node trust, most of them rely on a centralized manager, which is susceptible to single point of failure and being tampered with.

Therefore, with the decentralized and tamper-proof characteristics, the blockchain technique attracts more attentions to secure networks. In [19], a blockchain-based decentralized registration framework is designed to solve the problem of misconfiguration and malicious attacks. The authors propose a trusted network framework for flying ad-hoc networks, utilizing the blockchain technique to realize the distributed peer-topeer security [20]. The authors in [21] introduce a trusted self-organizing framework of UAV networks based on the blockchain to enhance the internal security. In [22], the authors present an efficient and secure data sharing approach for Internet of things via adopting the blockchain. [23] presents a private blockchain-enabled fine-grained access control mechanism for the software defined network environment. The authors in [24] propose a blockchain-based access control framework for industrial Internet of things. The above works illustrate that the blockchain technique enables to improve the network safety, and it is possible to employ the blockchain to ensure routing security in UAV networks.

As a significant issue, routing algorithms have been widely studied. For instance, the topology-based routing methods pose heavy communication overloads [25]. The greedy geographic routing methods are susceptible to falling into the local optimal solution [26]. The bio-inspired routing approaches struggle with parameter sensitivity and slow convergence [27]. Additionally, the mentioned algorithms are not suitable for distributed and highly dynamic UAV networks, where nodes can only obtain the localized information and topologies change constantly. Hence, it is significant to design a routing algorithm, effectively adapting to dynamics and local information. With the advance in reinforcement learning (RL) techniques, there exist a lot of works related to RL-based routing methods. For example, authors in [28] utilize Q-learning algorithms to learn routing decisions, and the method is further improved in [29], [30]. The authors in [31] propose a MADQN-based algorithm to overcome the interference and link instability caused by line-ofsight communication links. [29] proposes a distributed multiagent deep RL (DRL)-based routing solution to overcome link layer jamming attacks. Authors in [32] investigate the

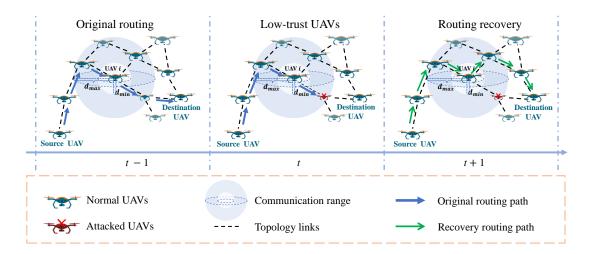


Fig. 1. Routing recovery in a time-varying UAV network.

MAPPO and multi-agent meta-PPO to optimize the network performances under both fixed and time-varying demands, respectively. As above, RL-based mechanisms can solve routing problems effectively. Nevertheless, most of the above works lack considerations of the security in a high dynamic and trustless UAV network with malicious nodes.

Hence, in this work, we consider utilizing the blockchain for improving the security of UAV networks with malicious nodes, and employ DRL techniques to deal with the trusted routing issues.

III. SYSTEM MODEL AND PROBLEM FORMULATION

A. Network Model

As show in Fig. 1, we depict the routing process in a time-varying UAV network, in which N nodes are distributed randomly, and each UAV holds a unique identity. Further, the UAV network is modeled as a weighted undirected graph $\mathcal{G}(\mathcal{U},\mathcal{E})$, where \mathcal{U} denotes the set of all UAVs, and $\mathcal{U}_{\rm s}$, $\mathcal{U}_{\rm r}$, and $\mathcal{U}_{\rm d}$ indicate the sets of source, relay, and destination UAVs, respectively. The values of $|\mathcal{U}_{\rm s}|$ and $|\mathcal{U}_{\rm d}|$ is equal to the number of demand set R. \mathcal{E} indicates the set of communication link status among UAVs. We utilize $e(i,j) \in \mathcal{E}$ to represent whether there exists a communicable link between UAVs i and j ($i \neq j, \forall i, j \in \mathcal{U}$). Specifically, e(i,j) = 1 indicates the link is effective, and e(i,j) = 0 denotes there exist no direct links between UAVs i and j.

In particular, the time period is divided into T time slots, and the length of each time slot is τ . With regard to demand $r \in R$, it is denoted as a tuple $\{s,d,\mathcal{L}^r\}$, where s,d, and \mathcal{L}^r indicate the source UAV, destination UAV, and data size of demand r, respectively. Moreover, $P_{sd}^r = (e(s,i),\cdots,e(j,d))$ indicates a complete routing path for transmitting demand r from source UAV $s \in \mathcal{U}_s$ to corresponding destination UAV $d \in \mathcal{U}_d$. In the UAV network model, for a specific demand, any two UAVs can serve as the source and destination nodes, while the remaining UAVs act as relay nodes to forward data.

It is noted that a relay node can have its own original demand to transmit and thus become a source node. It indicates that the roles of source, relay, and destination nodes are determined based on the specific data transmission requirements.

The location of UAV $i \in \mathcal{U}$ at time slot t is indicated by $\Theta_{i(t)} = (x_{i(t)}, y_{i(t)}, z_{i(t)})$ in three-dimensional Cartesian coordinates. The velocity in x, y, and z directions of UAV i is represented by $\boldsymbol{\nu}_{i(t)} = (\nu_{i(t)}^x, \nu_{i(t)}^y, \nu_{i(t)}^z, \nu_{i(t)}^z)$, and the velocities of all UAVs are denoted as $\boldsymbol{V}(t) = \{\boldsymbol{\nu}_{1(t)}, \boldsymbol{\nu}_{2(t)}, \cdots, \boldsymbol{\nu}_{N(t)}\}$. Specifically, at the beginning of each time slot t, the UAV is able to instantaneously adjust its flight direction and speed, and subsequently fixes the flight direction and maintains a constant speed. From a kinematic view, $\Theta_{i(t)}$ is related to $\boldsymbol{\nu}_{i(t)}$ of UAV i. Specially, if UAV i has a velocity of $\boldsymbol{\nu}_{i(t)}$ at the beginning of time slot t and maintains a uniform speed within the time slot, its location change can be approximated as $\Delta\Theta_{i(t)} \approx \boldsymbol{\nu}_{i(t)}\tau$. Therefore, at the next time slot, the location $\Theta_{i(t+1)}$ can be computed as $\Theta_{i(t+1)} = \Theta_{i(t)} + \boldsymbol{\nu}_{i(t)}\tau$.

For data transmission, the location of the UAV determines the establishment and maintenance of communication links. At time slot t, only when the Euclidean distance $d_{i(t)j(t)}$ between UAVs i and j is within the communication range, a communication link can potentially be established for data transmission. Here,

$$\frac{d_{i(t)j(t)} = \|\Theta_{i(t)} - \Theta_{j(t)}\|_{2}}{\sqrt{(x_{i(t)} - x_{j(t)})^{2} + (y_{i(t)} - y_{j(t)})^{2} + (z_{i(t)} - z_{j(t)})^{2}}},$$
(1)

and the distance between UAV i and possible connected UAV $k \in K_{i(t)}$ should satisfy

$$d_{i(t)k(t)} \leqslant d_{i,max}, \forall k \in K_{i(t)}, i \in \mathcal{U}, t \in T, \tag{2}$$

in which $d_{i,max}$ denotes the maximum communication distance of UAV i. $K_{i(t)}$ indicates the set of UAVs that satisfies the constraint in (2) for UAV i at time slot t. Additionally, $d_{i(t)j(t)}, \forall i, j \in \mathcal{U}$ should satisfy

$$d_{min} \leqslant d_{i(t)j(t)}, \forall i, j \in \mathcal{U}, t \in T.$$
 (3)

Here, d_{min} represents the safe distance to avoid collisions among UAVs.

However, the fact that UAV j is within the communication range of UAV i does not necessarily represent that e(i,j)=1. The reasons are that UAV j may be malicious, and the limited resources of UAV i can only support the communication with a certain number of UAVs. Specially, $\mathfrak q$ UAVs are selected to establish effective links, combining the trust value of UAVs and the distance from UAV i. The set of the $\mathfrak q$ UAVs is represented as $\Gamma_{i(t)}$, which is dynamically adjusted according to the network topology.

B. Attack Model

Due to the unsecure environment, UAVs may be vulnerable to attacks and become malicious nodes. Hence, it is necessary to model the attack and then analyze the impact of attacks on the network. Generally, most attacks destroy the network via specific standards, such as prioritizing attacks on important nodes. Therefore, we design a deliberate attack model based on the node ranking, in which UAVs with higher importance are attacked in priority [33], [34]. In this way, the proposed approach can be more effectively applied to defend against such attacks and reduce the total delay caused by the malicious activities.

Considering the node degree $\zeta_{i(t)}$ of UAV i and the weight $I_{i(t)j(t)}$ of link $e_{i,j}$, the importance $\Lambda_{i(t)}$ of UAV i at time slot t is calculated as

$$\Lambda_{i(t)} = \zeta_{i(t)} + \sum_{j \in \Gamma_{i(t)}} I_{i(t)j(t)} \cdot (1 - \frac{\zeta_{j(t)} - 1}{\zeta_{i(t)} + \zeta_{j(t)} - 2}), t \in T,$$
(4)

in which

$$\zeta_{i(t)} = \sum_{j \in \mathcal{U}} e(i, j), t \in T, \tag{5}$$

and

$$I_{i(t)j(t)} = Z \cdot \frac{2}{m+2}.\tag{6}$$

Wherein, $Z = (\zeta_{i(t)} - m - 1) \cdot (\zeta_{j(t)} - m - 1)$ indicates the connectivity ability of link $e_{i,j}$, and m represents the number of triangles containing link $e_{i,j}$ in the UAV network topology.

Besides, we utilize f and \mathcal{U}_f to denote the number and set of malicious UAVs, respectively. When malicious UAVs are recognized, they are isolated from the network, leading to the unavailability of original routing paths. Therefore, a new transmission path should be found for recovering the routing, as shown in Fig. 1.

C. Communication Model

The path loss $L_{i(t)j(t)}$ of the communication channel between UAVs i and j is approximated by a free space path loss model [35], which remains constant within time slot t, i.e.,

$$L_{i(t)j(t)} = 10^{\vartheta \lg d_{i(t)j(t)} + \lg(\frac{4\pi f}{c})^2}, \tag{7}$$

in which ϑ indicates the path loss exponent, f represents the carrier frequency, and c is the speed of light.

Meanwhile, the signal-to-noise ratio $\Psi_{i(t)j(t)}$ in the channel of UAVs i and j at time slot t is

$$\Psi_{i(t)j(t)} = \frac{\mathcal{P}_{i(t)j(t)}^{tr}}{\sigma_{i(t)j(t)}^{2} L_{i(t)j(t)}},$$
(8)

where $\mathcal{P}^{tr}_{i(t)j(t)}$ and $\sigma^2_{i(t)j(t)}$ represent the transmission power and noise power, respectively. Accordingly, based on Shannon theory [36], the transmission rate $G_{i(t)j(t)}$ from UAV i to UAV j at time slot t is

$$G_{i(t)j(t)} = B_{i(t)j(t)} \log_2(1 + \Psi_{i(t)j(t)}),$$
 (9)

where $B_{i(t)j(t)}$ indicates the channel bandwidth between UAVs i and j at time slot t. Besides, due to limited UAV-to-UAV communication resources, the total amount of demands passing $e(i,j) \in \mathcal{E}$ at time slot t should satisfy

$$\sum_{r \in R} \eta_{i(t)j(t)}^r \mathcal{L}^r \leqslant \tau G_{i(t)j(t)}, \forall i, j \in \mathcal{U}, t \in T,$$
 (10)

in which $\eta^r_{i(t)j(t)}$ is a binary variable denoting whether demand r passes the link e(i,j) at time slot t,

$$\eta^r_{i(t)j(t)} \!=\! \begin{cases} 1, & \text{if } r \text{ is transmitted via } e(i,j) \text{ in time slot } t, \\ 0, & \text{otherwise.} \end{cases}$$

D. Energy Consumption

The energy consumption of UAVs is composed of two parts: the communication energy consumption of receivers and transmitters, and the energy required for UAV movements [37], [38].

1) Communication Consumption: For UAV i, $E_{i(t)}^{re}$ denotes the total energy consumption in time slot t for receiving the demand $r \in R$ from another UAV j, i.e.,

$$E_{i(t)}^{re} = \sum_{r \in R} \sum_{i \in \mathcal{U}_{a} \cup \mathcal{U}_{r}} \mathcal{L}^{r} \cdot E_{\text{elec}} \cdot \eta_{j(t)i(t)}^{r}, \tag{12}$$

and the transmission energy $E_{i(t)}^{tr}$ is

$$E_{i(t)}^{tr} = \sum_{r \in R} \sum_{j \in \mathcal{U}_r \cup \mathcal{U}_d} \mathcal{L}^r \cdot (E_{\text{elec}} + \xi_{fs} \cdot d_{i(t)j(t)}^2) \cdot \eta_{i(t)j(t)}^r, \quad (13)$$

where \mathcal{L}^r and E_{elec} indicate the bit length of demand r and the energy consumption of radio electronic devices per bit, respectively [37]. ξ_{fs} represents the energy consumption factor of the power amplifier, corresponding to the proposed free space model in (7). Besides, $d_{i(t)j(t)}$ denotes the distance between UAVs i and j at time slot t. Then, the communication energy consumption of UAV i at time slot t is

$$E_{i(t)}^{com} = E_{i(t)}^{re} + E_{i(t)}^{tr}. (14)$$

2) Mobility Consumption: The mobility energy consumption enables UAVs to hover and move in horizontal and vertical directions, ensuring that UAVs can operate normally. According to [39], in time slot t, the mobility energy consumption model

can be represented as

$$E_{i(t)}^{mov} = \int_{0}^{\tau} \mathcal{P}_{i(t)}^{hov} dt + \frac{M_{i} \left(\boldsymbol{\nu}_{i(t+\tau)}^{2} - \boldsymbol{\nu}_{i(t+0)}^{2} \right)}{2} + g M_{i} \Delta z_{i(t)},$$
(15)

where M_i and g are the mass of UAV i and the acceleration of gravity, respectively. $\nu_{i(t)}$ is the speed of UAV i, and $\Delta z_{i(t)} =$ $z_{i(t+\tau)} - z_{i(t+0)}$ is the movement distance against gravity during time slot t. $\mathcal{P}_{i(t)}^{hov}$ represents the hover power for UAV i in the instantaneous time t, i.e.,

$$\mathcal{P}_{i(t)}^{hov} = \mathcal{P}_{b}^{hov} \left(1 + \frac{3\nu_{i(t)}^{2}}{U_{tip}^{2}} \right) + \\
\mathcal{P}_{I}^{hov} \left(\sqrt{\left(1 + \frac{\nu_{i(t)}^{4}}{4v_{0}^{4}} \right)} - \frac{\nu_{i(t)}^{2}}{2v_{0}^{4}} \right)^{\frac{1}{2}} + \frac{1}{2} \partial_{0} j_{0} i_{0} A_{0} \nu_{i(t)}^{3},$$
(16)

where \mathcal{P}_{b}^{hov} and \mathcal{P}_{I}^{hov} indicate the blade profile power and induced power at hovering states in [39], respectively. U_{tip} and v_0 denote the tip speed of rotor blades and average rotor induction speed hovering, respectively. ∂_0 , j_0 , i_0 , and A_0 represent the airframe drag ratio, air density, rotor solidity, and rotor disk area, respectively.

Consequently, the total energy consumption of UAV i at time slot t is

$$E_{i(t)}^{con} = E_{i(t)}^{com} + E_{i(t)}^{mov}. (17)$$

Besides, the total energy consumption cannot violate the energy capacity $E_{i(t)}^{max}$ of UAV i at time slot t, i.e.,

$$E_{i(t)}^{con} \le \beta E_{i(t)}^{max}, \forall i \in \mathcal{U}, t \in T,$$
 (18)

where β is the empirical power efficiency.

It is set within a rationale value, taking into account the recommended discharge of depth for the specific UAV battery and user safety requirements [40], [41].

E. Delay Model

The end-to-end routing delay mainly consists of the queue delay and transmission delay on the multi-hop path from the source UAV to destination UAV.

1) Queue Delay ($\mathcal{T}^{que,r}$): Since UAV networks are characterized by high mobility and unstable data traffic fluctuations, we introduce a queue buffer for each UAV to alleviate the network congestion. In particular, the amounts of demands received and transmitted by UAV i are denoted as $\varepsilon_{i(t)}^{re}$ and $\varepsilon_{i(t)}^{tr}$, respectively, i.e.,

$$\begin{cases} \varepsilon_{i(t)}^{re} = \sum_{r \in R} \sum_{j \in \mathcal{U}_s \cup \mathcal{U}_r} \eta_{j(t)i(t)}^r, \\ \varepsilon_{i(t)}^{tr} = \sum_{r \in R} \sum_{j \in \mathcal{U}_r \cup \mathcal{U}_d} \eta_{i(t)j(t)}^r, \end{cases} \quad t \in T.$$
 (19)

slot t-1 to t of UAV i is equal to the difference between the

numbers of received demands $\varepsilon_{i(t-1)}^{re}$ and transmitted demands $\varepsilon_{i(t-1)}^{tr}$ at time slot t-1, i.e.,

$$\Delta C_{i(t)} = \varepsilon_{i(t-1)}^{re} - \varepsilon_{i(t-1)}^{tr}.$$
 (20)

Then, for UAV i, C_i^{max} and $C_{i(t)}$ denote the maximum queue capacity and the total amount of queued demands (i.e., the queue length), respectively, which should satisfy

$$\begin{cases}
0 \le C_{i(t)} \le C_i^{max}, \\
0 \le \varepsilon_{i(t)}^{re} \le C_i^{max}, \\
0 \le \varepsilon_{i(t)}^{tr} \le C_{i(t)}.
\end{cases}$$
(21)

Additionally, $C_{i(t)}$ is calculated by adding $\Delta C_{i(t)}$ to the queue length at time slot t-1, i.e.,

$$C_{i(t)} = C_{i(t-1)} + \Delta C_{i(t)}.$$
 (22)

Based on the above analysis, when there exists demand $r \in R$ arriving at UAV i with $C_{i(t)}$, the queue delay of demand r for transmitting to next-hop UAV j is

$$\mathcal{T}_{i(t)}^{que,r} = \sum_{r \in \mathcal{C}_{i(t)}} \frac{\eta_{i(t)k(t)}^r \mathcal{L}^r}{G_{i(t)k(t)}}, k \in \Gamma_{i(t)}, \tag{23}$$

where $C_{i(t)}$ is the queued demand set of UAV i at time slot t, and $|C_{i(t)}| = C_{i(t)}$. Besides, to maintain the orderly processing of demands, the queuing model leverages first-in-first-out rules.

2) Transmission Delay $(\mathcal{T}^{tr,r})$: The one-hop transmission delay of demand r from UAV i to UAV j at time slot t is

$$\mathcal{T}_{i(t)j(t)}^{tr,r} = \frac{\mathcal{L}^r}{G_{i(t)j(t)}} \eta_{i(t)j(t)}^r, r \in R.$$
 (24)

Hence, the sum of one-hop delay for transmitting demand rfrom UAV i to UAV j at time slot t is

$$\mathcal{T}_{i(t)j(t)}^{r} = \mathcal{T}_{i(t)j(t)}^{tr,r} + \mathcal{T}_{i(t)j(t)}^{que,r}, t \in T,$$
 (25)

in which the value of $\mathcal{T}^r_{i(t)j(t)}$ cannot exceed the maximum tolerated delay \mathcal{T}^{max}_{one} for one-hop, i.e.,

$$\mathcal{T}_{i(t)j(t)}^r \le \mathcal{T}_{one}^{max}.\tag{26}$$

When the routing path P_{sd}^r for transmitting demand r from source UAV s to destination UAV d is determined, the end-toend delay \mathcal{T}^r can be calculated as

$$\mathcal{T}^r = \sum_{t \in Te(i,j) \in \mathsf{P}^r_{\mathsf{sd}}} \mathcal{T}^r_{i(t)}. \tag{27}$$

F. Problem Formulation

1) Flow Constraints: The demand r from source UAV $s \in$ U_s can only be received by one another UAV j, i.e.,

$$\sum_{j \in \mathcal{U}_{\mathbf{r}} \cup \mathcal{U}_{\mathbf{d}}} \eta_{\mathbf{s}(t)j(t)}^r = 1, \forall r \in R, t \in T.$$
 (28)

Further, the transformation of queue length $\Delta C_{i(t)}$ from time Recall that $\eta^r_{s(t)j(t)} \in \{0,1\}$ denotes whether demand r via link $e(s,j) \in \mathcal{E}$, 1 if passing and 0 otherwise. With respect to a middle UAV $j \in \mathcal{U}_r$, the flow conservation should be satisfied:

$$\begin{cases}
\sum_{i \in \mathcal{U}_{s} \cup \mathcal{U}_{r}} \eta_{i(t)j(t)}^{r} = \sum_{i \in \mathcal{U}_{r} \cup \mathcal{U}_{d}} \eta_{j(t)i(t)}^{r} + \eta_{j(t)j(t+1)}^{r}, \\
\forall r \in R, j \in \mathcal{U}_{r}, t = 1, \\
\sum_{i \in \mathcal{U}_{s} \cup \mathcal{U}_{r}} \eta_{i(t)j(t)}^{r} + \eta_{j(t-1)j(t)}^{r} = \sum_{i \in \mathcal{U}_{r} \cup \mathcal{U}_{d}} \eta_{j(t)i(t)}^{r} + \\
\eta_{j(t)j(t+1)}^{r}, \forall r \in R, j \in \mathcal{U}_{r}, t \in \{2, \dots, T-1\}, \\
\sum_{i \in \mathcal{U}_{s} \cup \mathcal{U}_{r}} \eta_{i(t)j(t)}^{r} + \eta_{j(t-1)j(t)}^{r} = \sum_{i \in \mathcal{U}_{r} \cup \mathcal{U}_{d}} \eta_{j(t)i(t)}^{r}, \\
\forall r \in R, i \in \mathcal{U}, t = T
\end{cases} (29)$$

Besides, a demand can select only one routing path, i.e.,

$$\begin{cases}
\sum_{j \in \mathcal{U}_{\mathbf{r}} \cup \mathcal{U}_{\mathbf{d}}} \eta_{i(t)j(t)}^{r} + \eta_{i(t)i(t+1)}^{r} = 1, \\
\forall r \in R, i \in \mathcal{U}_{\mathbf{r}}, t \in \{1, \dots, T-1\}, \\
\sum_{j \in \mathcal{U}_{\mathbf{r}} \cup \mathcal{U}_{\mathbf{d}}} \eta_{i(t)j(t)}^{r} = 1, \forall r \in R, i \in \mathcal{U}_{\mathbf{r}}, t = T.
\end{cases} (30)$$

Similar to the source UAVs, the destination UAV $d \in \mathcal{U}_d$ can only receive demand r from one another UAV i, i.e.,

$$\sum_{i \in \mathcal{U}_{s} \cup \mathcal{U}_{r}} \eta_{i(t)d(t)}^{r} = 1, \forall r \in R, t \in T.$$
(31)

2) Problem Formulation: The objective is to minimize the total end-to-end delay of networks with attacked UAVs that may become malicious nodes, and the corresponding optimization problem is formulated as

$$\begin{split} \mathscr{P}0: & \min_{\pmb{\eta}} \sum_{r \in R} \mathcal{T}^r \\ & \text{s.t. } (3), (2), (10), (18), (21), (26), (28) - (31), \qquad (32) \\ & \eta^r_{i(t)j(t)} \in \{0, 1\}, \\ & \forall t \in T, e(i, j) \in \mathcal{E}, i, j \in \mathcal{U}, i, j \notin \mathcal{U}_{\mathtt{f}}, r \in R, \end{split}$$

where $\eta = \{\eta^r_{i(t)j(t)}, \forall t \in T, e(i,j) \in \mathcal{E}, i,j \in \mathcal{U}, i,j \notin \mathcal{U}_{\mathbf{f}}, r \in R\}$, in which $i,j \notin \mathcal{U}_{\mathbf{f}}$ represents that demand $r \in R$ avoids being transmitted by malicious UAVs in the network. It is observed that $\mathscr{P}0$ is in the form of ILP and NP-hard to deal with [42]. Therefore, in the next section we propose the feasible solutions.

IV. ALGORITHM DESIGN

In this section, to realize the security in the UAV network with malicious nodes, the BTMM is proposed. Further, for effectively solving $\mathcal{P}0$, we reformulate it into a Dec-POMDP and propose an MADDQN-based algorithm to make dynamic routing decisions.

A. Blockchain-based Trust Management Mechanism

Distributed trust mechanisms enable UAVs to autonomously evaluate the reliability of neighbor UAVs without relying on a central authority. This approach enhances the scalability and fault tolerance, especially for dynamic UAV networks [43].

Since the consortium blockchain is decentralized and jointly maintained by multiple pre-selected nodes, it can be applied for UAVs to create the secure and reliable communication network. In detail, the malicious behaviors of UAVs are analyzed, and we design a mechanism to evaluate the trust value of UAV nodes. Besides, a consensus UAV update mechanism is proposed to improve the security of the PBFT consensus algorithm, which isolates UAVs with trust values below a safety threshold.

- 1) Node Trust Evaluation Mechanism: The malicious behaviors of UAVs are analyzed in two aspects. In particular, when the malicious UAVs receive the demand, they may drop it or do not transmit it following the specific routing path. Hence, considering the delivery rate and the correctness of transmission paths, the trust evaluation mechanism for each UAV is designed as follows.
 - Delivery rate evaluation: The value of the delivery rate is applied to evaluate the malicious UAVs with behaviors rejecting to transmit received demands, which is represented as the ratio of total transmitted demands $\sum_{k=1}^{t} \varepsilon_{i(k)}^{tr}$ to total received demands $\sum_{k=1}^{t} \varepsilon_{i(k)}^{re}$ by UAV i at time slot t, i.e.,

$$\mathbb{T}_{i(t)}^{dr} = \frac{\sum_{k=1}^{t} \varepsilon_{i(k)}^{tr}}{\sum_{k=1}^{t} \varepsilon_{i(k)}^{re}}, \forall i \in \mathcal{U}, \sum_{k=1}^{t} \varepsilon_{i(k)}^{re} > 0, t \in T.$$
 (33)

Besides, $\sum_{k=1}^{t} \varepsilon_{i(k)}^{re} = 0$ denotes UAV i does not receive demands, and $\mathbb{T}_{i(t)}^{dr}$ remains the initial value.

• Transmission path evaluation: The malicious UAVs typically do not follow the specified path for transmitting demands to the next-hop UAV, which results in increasing transmission delay [44]. Therefore, we evaluate the correctness of transmission paths to partly assess whether the UAV is malicious. When UAV j receives a demand r, it checks whether the data is transmitted by UAV i following specified paths. If not, the normal UAV j records the relevant abnormal information of UAV i and then uploads it to the consensus UAVs as a basis for calculating node trust values. When the UAV is not recorded, it is defaulted as a safe node. In particular, we can calculate the evaluation value of UAV i in terms of the transmission path at time slot t by

$$\mathbb{T}_{i(t)}^{tp} = 1 - \frac{\sum_{k=1}^{t} \mathfrak{B}_{i(k)}}{\sum_{k=1}^{t} \mathbb{B}_{i(k)}}, \forall i \in \mathcal{U}, \sum_{k=1}^{t} \mathbb{B}_{i(k)} > 0, t \in T, \quad (34)$$

where $\sum_{k=1}^t \mathfrak{B}_{i(t)}$ and $\sum_{k=1}^t \mathbb{B}_{i(t)}$ indicate the total numbers of non-specified and total transmission paths of UAV i at time slot t, respectively. In addition, when $\sum_{k=1}^t \mathbb{B}_{i(t)} = 0$, $\mathbb{T}_{i(t)}^{tp}$ remains unchanged at the initial value.

Considering the above two factors, the comprehensive trust

Algorithm 1 Adaptive Weights based Trust Evaluation Process

1: **Initialization:** $\mathbb{T}_{i(1)}=1, \mathbb{T}_{i(1)}^{dr}=1, \text{and } \mathbb{T}_{i(1)}^{tp}=1, \forall i \in \{1,...,N\}$. Initialize the values of the demand delivery probability p_1 and the correct path selection probability p_2 , respectively.

```
2:
      for each time step t do
            for each UAV i do
 3:
                Update \mathbb{T}^{dr}_{i(t)} via (33).

Update \mathbb{T}^{tp}_{i(t)} via (34).

Calculate \psi^0_{i(t)}, \psi^1_{i(t)}, and \psi^2_{i(t)} by (36).

Compute \mathbb{T}_{i(t)} according to (35).
 4:
 5:
 6:
 7:
                 if \mathbb{T}_{i(t)} < \mathbb{T}_{thr} then
 8:
                      Flag UAV i as malicious.
 9:
10:
                 end if
            end for
11:
12: end for
```

value of UAV i at time slot t+1 is calculated as

$$\mathbb{T}_{i(t+1)} = \psi_{i(t)}^{0} \mathbb{T}_{i(t)} + \psi_{i(t)}^{1} \mathbb{T}_{i(t)}^{dr} + \psi_{i(t)}^{2} \mathbb{T}_{i(t)}^{tp}, \\ \forall i \in \mathcal{U}, t \in T.$$
 (35)

Wherein, $\mathbb{T}_{i(1)}=1$, $\mathbb{T}^{dr}_{i(1)}=1$, and $\mathbb{T}^{tp}_{i(1)}=1$ indicate the initial trust values for UAV i. $\psi^0_{i(t)}$, $\psi^1_{i(t)}$, and $\psi^2_{i(t)}$ are hyperparameters, denoting the weights of $\mathbb{T}_{i(t)}$, $\mathbb{T}^{dr}_{i(t)}$, and $\mathbb{T}^{tp}_{i(t)}$, respectively, and $\psi^0_{i(t)}+\psi^2_{i(t)}+\psi^2_{i(t)}=1$.

Moreover, considering that $\mathbb{T}^{dr}_{i(t)}$ and $\mathbb{T}^{tp}_{i(t)}$ are directly related to the trust value evaluation, we propose the adaptive method to determine the weights of trust values, detailed as

$$\begin{cases} \psi_{i(t)}^{0} = 0.5 \times \frac{\mathbb{T}_{thr}}{\mathbb{T}_{i(t)}}, \\ \psi_{i(t)}^{1} = \frac{(1 - \psi_{i(t)}^{0}) \cdot (1 - \mathbb{T}_{i(t)}^{dr})}{2 - \mathbb{T}_{i(t)}^{dr} + \mathbb{T}_{i(t)}^{tp}}, 0 \leq \mathbb{T}_{i(t)}^{dr} + \mathbb{T}_{i(t)}^{tp} \leq 2, \\ \psi_{i(t)}^{2} = \frac{(1 - \psi_{i(t)}^{0}) \cdot (1 - \mathbb{T}_{i(t)}^{tp})}{2 - \mathbb{T}_{i(t)}^{dr} + \mathbb{T}_{i(t)}^{tp}}. \end{cases}$$
(36)

When $\mathbb{T}^{dr}_{i(t)}+\mathbb{T}^{tp}_{i(t)}=2$, $\psi^1_{i(t)}=\psi^2_{i(t)}=\frac{(1-\psi^0_{i(t)})}{2}$. \mathbb{T}_{thr} can be determined by the security requirements [45].

Algorithm 1 indicates the trust evaluation process of the adaptive weight method. For each UAV i, initialize $\mathbb{T}_{i(1)}=1,\mathbb{T}^{dr}_{i(1)}=1,$ and $\mathbb{T}^{tp}_{i(1)}=1$ as 1. Additionally, initialize the probabilities associated with the behaviors of malicious UAVs. Specifically, the demand delivery probability and the correct path selection probability are denoted as p_1 and p_2 , respectively. At each time step t, update $\mathbb{T}^{dr}_{i(t)}$ via (33) and $\mathbb{T}^{tp}_{i(t)}$ via (34). Calculate $\psi^0_{i(t)}, \psi^1_{i(t)}$, and $\psi^2_{i(t)}$ according to (36). Then, compute $\mathbb{T}_{i(t)}$ based on (35). If $\mathbb{T}_{i(t)}$ is less than \mathbb{T}_{thr} , flag UAV i as malicious.

2) Trust-based PBFT Algorithm: To reach the consensus in distributed UAV networks with malicious nodes, we introduce the PBFT algorithm, which can tolerate less than 1/3 fault nodes [46]. In this work, to further ensure the security of

PBFT processes, we propose the trust-based PBFT (TPBFT) algorithm, which updates the consensus UAV set \mathcal{U}_c , as illustrated in Algorithm 2. In detail, to initialize the network, we select 3n+1 UAVs with the largest trust values as consensus node $c \in \mathcal{U}_c$, in which the UAV with highest trust value is selected as the leader, and the candidate consensus UAV set is denoted as $\mathcal{U}_{\bar{c}}$ (line 1). With \mathcal{U}_c identified, the consensus process for transactions in Fig. 2 is performed and primarily includes three stages: *pre-prepare*, *prepare*, and *commit* (line 3). In particular, the non-consensus UAVs upload transactions with recorded information to the nearest consensus UAV, which relays it to the leader consensus UAV for verification, and then the leader starts the three-stage processes among all consensus UAVs as follows.

- Firstly, the leader packs the validated transaction into a block and then broadcasts the *pre-prepare* message $\langle\langle pre-prepare, \mathfrak{k}, \mathfrak{n}, \mathfrak{d}\rangle_{\mathfrak{a}_\mathfrak{p}}, \mathfrak{m}\rangle$ to non-leader consensus UAVs. Wherein, \mathfrak{k} is the current view, \mathfrak{n} is the sequence number, \mathfrak{m} is the original request, and the digest of \mathfrak{m} is \mathfrak{d} . The leader attaches its digital signature $\mathfrak{a}_\mathfrak{p}$ and message authentication codes (MACs) to ensure the authenticity.
- In the *prepare* stages, each non-leader consensus UAV verifies the received *pre-prepare* message by checking the signature and MACs of the leader and the consistency of \mathfrak{d} with \mathfrak{m} . After successful verification, consensus UAV c broadcasts a *prepare* message $\langle prepare, \mathfrak{k}, \mathfrak{n}, \mathfrak{d}, \mathfrak{c} \rangle_{\mathfrak{a}_c}$ to other consensus UAVs. Furthermore, each consensus UAV compares the received *prepare* messages with the original *pre-prepare* message to ensure the consistency of ($\mathfrak{k}, \mathfrak{n}, \mathfrak{d}$). The consensus requires more than 2n valid *prepare* signatures, where n is the maximum number of faulty consensus UAVs.
- In the *commit* phase, if the consensus UAV receives more than or equal to 2n valid *prepare* messages, they broadcast *commit* messages to the other consensus UAVs for verification. If each consensus UAV receives 2n+1 *commit* messages, the consensus is reached.

Further, in Algorithm 2, if \mathcal{M} rounds of consensus processes are finished, the process for updating \mathcal{U}_c is performed (line 4). If there exists UAV $c \in \mathcal{U}_c$ with a lower trust value than the safety threshold \mathbb{T}_{thr} , it is removed from the \mathcal{U}_c (lines 5-7). Otherwise, the consensus UAV c with the minimum trust value is removed from \mathcal{U}_c (lines 8-10). Then, a new consensus UAV is invited from the candidate set $\mathcal{U}_{\bar{c}}$ (line 11). Fig. 2 illustrates the node update process when there exists one (f = 1) malicious consensus UAV, detailed as follows.

- Remove malicious consensus UAV é: The consensus leader UAV launches the request for updating the set U_c and sends the information of the consensus UAV with lower trust values (i.e., UAV é) than T_{thr} to the other UAVs in U_c. In both the launching and exchanging phases, when consensus UAV c ∈ U_{c\é} receives 2f messages that UAV é is untrusted, it is deleted from consensus set U_c.
- Invite candidate consensus UAV ce: After removing UAV ce, the other consensus UAVs send an invitation message

Algorithm 2 TPBFT Algorithm

value $\mathbb{T}_{\tilde{c}(t)}$ to \mathcal{U}_{c} .

end if

13: end for

12:

```
Input: UAV set \mathcal{U}, security threshold \mathbb{T}_{thr}, and initial trust
     value \mathbb{T}_{i(1)} for all i \in \mathcal{U}.
Output: \mathcal{U} and consensus UAV set \mathcal{U}_c.
 1: Initialization: \mathcal{U}, \mathbb{T}_{i(1)}, \mathbb{T}_{thr}, \mathcal{U}_c, and candidate consensus
     UAV set \mathcal{U}_{\tilde{c}}.
 2: for t \in T do
        The non-consensus UAVs upload transactions for verifi-
        cation, and the consensus is reached among all c \in \mathcal{U}_c
        via the three-stage processes illustrated in Fig. 2.
        if \mathcal{M} rounds of consensus processes are finished then
 4:
            if \exists c \in \mathcal{U}_c, \mathbb{T}_{c(t)} < \mathbb{T}_{thr} then
 5:
                Delete UAV c from \mathcal{U}_c.
 6:
            end if
 7:
            if \forall \mathtt{c} \in \mathcal{U}_{\mathtt{c}}, \ \mathbb{T}_{\mathtt{c}(t)} > \mathbb{T}_{thr} then
 8:
                Delete consensus UAV c with the minimum trust
 9:
                value from \mathcal{U}_c.
            end if
10:
            Invite candidate UAV \tilde{c} \in \mathcal{U}_{\tilde{c}} with the maximum trust
11:
```

to a non-consensus UAV \tilde{c} with the highest trust value as the candidate. When UAV \tilde{c} receives 2f+1 invitation messages, it sends a reply message to UAV $c\in\mathcal{U}_c$ for the acceptance of invitation.

• Update consensus UAV \tilde{c} : When consensus UAVs in \mathcal{U}_c receive 2f+1 reply messages, they send the update of UAV \tilde{c} to the other consensus UAVs and inform \tilde{c} that it is updated as a formal consensus UAV of \mathcal{U}_c . Therefore, the consensus node update is finished.

According to the verified transaction, consensus UAVs recalculate the trust value based on the designed node trust evaluation mechanism, and isolate the UAVs with values below the safety threshold, to ensure the network security for transmitting demands. It is noted that trust values are managed by the selected consensus UAVs, which can avoid the single-node fault and tolerate 1/3 malicious routing nodes to construct trusted networks. Besides, when the malicious UAV is removed from the network, the topology variation puts forward the requirement for dynamic routing solutions. Accordingly, we further propose the DRL-based method to deal with $\mathcal{P}0$, which is reformulated based on the Dec-POMDP.

B. Dec-POMDP Reformulation for the Trusted Routing Problem

Based on the trusted environment, since each UAV can only obtain local information from its neighbor UAVs for the distributed network, we reformulate $\mathcal{P}0$ into a Dec-POMDP, which mainly consists of five parts: agent set \mathcal{N} , state space \mathcal{S} , action space \mathcal{A} , reward function \mathcal{R} , and discount factor γ .

• Agent set: $\mathcal{N} = \{1, \dots, N\}$ represents the set of agents (all UAVs).

• State space: At the start of each time slot t, each agent $i \in \mathcal{U}$ gathers and obtains its own observation $o_{i(t)}$ from the environment. It is noted that each UAV can only observe the information of neighbor node $k \in \Gamma_{i(t)}$. Hence, the observation of agent i is

$$o_{i(t)} = \{\Theta_{i(t)}, \mathcal{C}_{i(t)}, \Theta_{k(t)}, \mathcal{C}_{k(t)}, E_{k(t)}^{con}, \mathbb{T}_{k(t)}\},$$

$$\forall i \in \mathcal{U}, k \in \Gamma_{i(t)}, t \in T,$$

$$(37)$$

in which $\Theta_{i(t)}$ and $\mathcal{C}_{i(t)}$ indicate the location and the queued demand set of UAV i, respectively. $\Theta_{k(t)}$, $\mathcal{C}_{k(t)}$, $E_{k(t)}^{con}$ and $\mathbb{T}_{k(t)}$ represent the location, queued demand set, energy consumption, and trust value of agent k at time slot t, respectively. The observations of all agents are obtained to aggregate the joint state s_t in time slot t, denoted as

$$\mathbf{s}_t = \{o_{1(t)}, o_{2(t)}, \cdots, o_{N(t)}\}, t \in T, \tag{38}$$

and the state space is indicated as $S = \{s_t, t \in T\}$.

As introduced in Section III-A, the number of UAVs in $k \in \Gamma_{i(t)}$ does not exceed q, and these UAVs are dynamically adjusted with the network variation. Additionally, the observation dimension of one neighbor UAV is fixed according to (37). Hence, when there exist UAVs joining or leaving, the dimension of observation $o_{i(t)}$ is stabilized via reselecting UAVs or performing zero-padding operations on the remaining dimensions.

• Action space: According to observation $o_{i(t)}$ at time slot t, the action set of agent i is denoted as

$$a_{i(t)} = \{a_{i(t)}^k\}, k \in \Gamma_{i(t)}, t \in T,$$
 (39)

where $a_{i(t)}^k$ represents that the next step of UAV i is UAV k at time slot t. Besides, the joint action of N agents at time slot t is

$$\mathbf{a}_t = \{a_{1(t)}, a_{2(t)}, \cdots, a_{N(t)}\}, t \in T.$$
 (40)

Correspondingly, the joint action space is indicated as $A = \{a_t, t \in T\}$.

Reward: R_{i(t+1)} indicates the instantaneous reward of agent i performing action a_{i(t)} at time slot t. The objective in \$\mathscr{P}\$0 is to minimize the total end-to-end delay cost of all demands, for which all agents cooperate to maximize their contributions, and then each agent obtains the same reward R_{t+1}. Therefore, the instantaneous reward is defined as the negative value with respect to the delay, i.e.,

$$\mathcal{R}_{t+1} = \mathcal{R}_{i(t+1)} = -10 \times \sum_{r \in R} \sum_{i,j \in \mathcal{U}} \min(\mathcal{T}_{i(t)j(t)}^r, \mathcal{T}_{one}^{max}), t \in T,$$

$$(41)$$

where $\mathcal{T}_{i(t),j(t)}^r$ and \mathcal{T}_{one}^{max} are shown in (25) and (26), respectively.

• Discount factor: $\gamma_i \in (0,1)$ is the discount factor of agent i for calculating the cumulative reward. $\gamma = \{\gamma_i, i \in \mathcal{U}\}$ is the discount factor set of all agents. A larger value of γ indicates the decision focusing on the long-term reward.

The policy π_i leads agent i to select an action $a_{i(t)}$ under

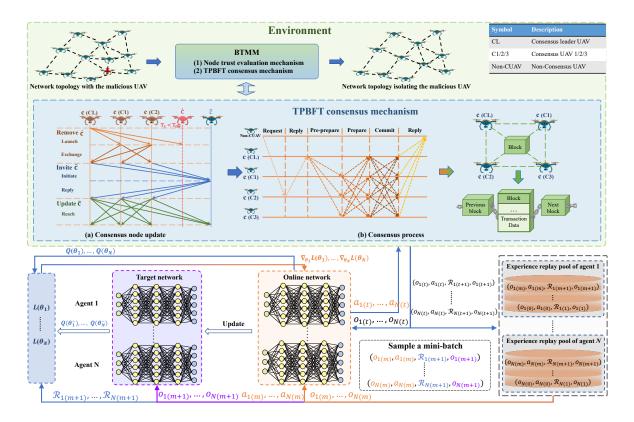


Fig. 2. BTMM-MADDQN algorithm framework for trusted routing in the UAV network.

observation $o_{i(t)}$. $\pi = \{\pi_1, \pi_2, \cdots, \pi_N\}$ indicates the joint policy of all agents. According to the Dec-POMDP and a specific policy π , we can obtain the routing paths. Hence, $\mathcal{P}0$ is transformed to find the optimal policy π^* , and the MADDQN-based intelligent approach is designed to obtain π^* for minimizing the total end-to-end delay of routing paths.

C. MADDQN Algorithm

The MADDQN is a Q-learning-based algorithm, which mainly constructs and maintains a Q-table containing the values of all state-action pairs. Formally, the Q-value function is defined as the expected total reward for taking action $a_{i(t)}$ in observation $o_{i(t)}$ for agent i at time slot t, i.e.,

$$Q(o_{i(t)}, a_{i(t)}) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma_i^k \mathcal{R}_{i(t+k+1)} \mid o_{i(t)} = o, a_{i(t)} = a\right], (42)$$

where $\mathbb{E}[\cdot]$ denotes the expected value. Through the Q-table, the optimal policy π_i^{\star} can be obtained by agent i in the process of continuous interactions with the environment. In time slot t, the policy π_i^{\star} is

$$\pi_i^* = \arg\max_{\pi} Q^*(o_{i(t)}, a_{i(t)}).$$
 (43)

Further, the following formula iteratively updates the Q-value function $Q(o_{i(t)}, a_{i(t)})$ via Bellman equations, i.e.,

$$Q(o_{i(t)}, a_{i(t)}) \leftarrow (1 - \alpha_i)Q(o_{i(t)}, a_{i(t)}) +$$

$$\alpha_i \left[\mathcal{R}_{i(t+1)} + \gamma_i \max_{a'_{i(t)}} Q(o_{i(t+1)}, a'_{i(t)}) \right], \quad (44)$$

where α_i is the learning rate that controls the size of update steps. $Q(o_{i(t+1)}, a'_{i(t)})$ is the Q-value function of agent i for taking action $a'_{i(t)}$ under the next observation $o_{i(t+1)}$, which is transformed by performing action $a_{i(t)}$ in current observation $o_{i(t)}$ with reward $\mathcal{R}_{i(t+1)}$ in time slot t. The term $y = \mathcal{R}_{i(t+1)} + \gamma_i \max_{a'_{i(t)}} Q(o_{i(t+1)}, a'_{i(t)})$ represents the Q-target.

However, maintaining the Q-table becomes exceptionally complex, when the dimensions of the state space and action space are large. Hence, the DQN algorithm is proposed to approximate Q-value functions via deep neural networks (DNNs) with the observation as inputs and the action as outputs [47]. There exist two DNNs in DQN algorithms, i.e., the online network with parameter θ_i and the target network with parameter θ_i^- of each agent i. By constantly updating the weight θ_i of online networks, the loss function $L(\theta_i)$ of agent i is minimized and typically computed as the mean squared error between Q-value function $Q(o_{i(t)}, a_{i(t)}; \theta_i)$ and Q-target $y_{i(t)}^{\text{DQN}}$ via randomly sampling B historical experience transitions from \mathcal{B}_i , i.e.,

$$L(\theta_i) = \mathbb{E}_{\mathcal{B}_i \sim \varpi} \left[\left(y_{i(t)}^{\text{DQN}} - Q(o_{i(t)}, a_{i(t)}; \theta_i) \right)^2 \right], \tag{45}$$

and $y_{i(t)}^{DQN}$ is

$$y_{i(t)}^{\text{DQN}} = \mathcal{R}_{i(t+1)} + \gamma_i \max_{a'_{i(t)}} Q(o_{i(t+1)}, a'_{i(t)}; \theta_i^-), \quad (46)$$

Algorithm 3 MADDQN-based Routing Algorithm

```
Input: \mathcal{U}, R, \alpha, \gamma, and \mathcal{E}.
Output: Optimal policy \pi^*.
 1: Initialization: Initialize the UAV network environment,
    hyper-parameters, and experience replay pool set \mathcal{B}.
 2: for each episode do
       for t = 1, \ldots, T do
 3:
          for i = 1, \ldots, N do
 4:
             The observation of agent i is set as o_{i(t)}.
 5:
             Agent i selects an action a_{i(t)} under observation
 6:
             o_{i(t)} using an \epsilon-greedy policy.
             Execute a_{i(t)}, and obtain reward \mathcal{R}_{i(t+1)} and next
 7:
             observation o_{i(t+1)}.
             Store the transition (o_{i(t)}, a_{i(t)}, \mathcal{R}_{i(t+1)}, o_{i(t+1)}) in the
 8:
             experience replay pool \mathcal{B}_i.
 9:
             Set o_{i(t)} \leftarrow o_{i(t+1)}.
             if |\mathcal{B}_i| > B then
10:
                Randomly select B samples from the experience
11:
                replay pool \mathcal{B}_i.
             end if
12:
             Compute the Q-target value y_{i(t)}^{\text{DDQN}} for each agent i
13:
             via (47).
14:
             Perform the gradient descent to update \theta_i by (48).
             Periodically update the target network parameter \theta_i^-
15:
             via (49) every W steps.
          end for
16:
       end for
17:
18: end for
```

where tuple $\varpi = (o_{i(t)}, a_{i(t)}, \mathcal{R}_{i(t+1)}, o_{i(t+1)})$ is a transition data stored in replay pool \mathcal{B}_i , which is designed to store the historical experience of agent i. The set of experience replay pools for all agents is denoted as $\mathcal{B} = \{\mathcal{B}_i, i \in \mathcal{U}\}$.

Nevertheless, the DQN-based algorithm may cause a large deviation, due to overestimating the Q-target value. To avoid the overestimation, a DDQN-based algorithm is leveraged by combining double Q-learning with the DQN, and decouples the action selection and calculation of Q-target values. It is theoretically proved that the DDQN-based approach can avoid the overestimation [48]. Correspondingly, Q-target $y_{i(t)}^{\text{DQN}}$ in loss function (45) is replaced by $y_{i(t)}^{\text{DDQN}}$, which is

$$y_{i(t)}^{\text{DDQN}} = \mathcal{R}_{i(t+1)} + \gamma Q(o_{i(t+1)}, \arg \max_{a'_{i(t)}} Q(o_{i(t+1)}, a'_{i(t)}; \theta_i) | \theta_i^-).$$
 (47)

Besides, the gradient of loss functions is denoted as $\nabla_{\theta_i} L(\theta_i)$, which is leveraged to update θ_i via the gradient descent, i.e.,

$$\theta_i \leftarrow \theta_i - \alpha_i \nabla_{\theta_i} L(\theta_i).$$
 (48)

Every \mathcal{W} steps, parameter θ_i^- is periodically updated to match the parameter of online network θ_i for stabilizing training and improving the convergence, i.e.,

$$\theta_i^- \leftarrow \theta_i.$$
 (49)

TABLE I PARAMETER SETTING

Parameter	Value	Parameter	Value
ϑ	2	f	2.4GHz
c	3×10^8 m/s	σ_{ij}^2	-110dBm
B_{ij}	2MHz	M_i	2kg
g	9.8m/s ²	$ u_i $	3m/s
\mathcal{P}_b^{hov}	9.1827W	\mathcal{P}_{I}^{hov}	11.5274W
U_{tip}	60m/s	v_0	2.4868m/s
∂_0	0.5017	J 0	1.205kg/m ³
\imath_0	0.0832	A_0	0.2827m^2
β	0.7	C_i^{max}	50

The detail of the proposed MADDQN is shown in Algorithm 3. Firstly, the algorithm initializes the UAV network environment, hyper-parameters, and experience replay pool set \mathcal{B} (line 1). At the beginning of each episode, the observation of each agent i is initialized as $o_{i(t)}$ (lines 2-5). Based on $o_{i(t)}$, each agent i selects an action $a_{i(t)}$ by leveraging an ϵ greedy policy, and then executes $a_{i(t)}$ to obtain reward $\mathcal{R}_{i(t+1)}$ and next observation $o_{i(t+1)}$ (lines 6-7). Hence, the agent iobtains transition $(o_{i(t)}, a_{i(t)}, \mathcal{R}_{i(t+1)}, o_{i(t+1)})$ and stores it into experience replay pool \mathcal{B}_i (line 8). Besides, the observation $o_{i(t)}$ of agent i is correspondingly updated by $o_{i(t+1)}$ (line 9). If the number of transition tuples in \mathcal{B}_i is larger than the minibatch B, B samples are randomly selected to calculate Q-target value $y_{i(t)}^{\scriptscriptstyle \mathrm{DDQN}}$ (lines 10-13). Additionally, the network performs a gradient descent step to update θ_i (line 14). The target network parameter θ_i^- is periodically updated according to (49) every W steps (line 15).

D. Complexity Analysis

Suppose that the width of the i-th layer of the neural network is W_i , and there exist M layers in total. S and A are the numbers of dimensions in the state space and the action space for an agent, respectively. The computational complexity from the i-th layer to the (i+1)-th layer involves the multiplication and summation of connection weights between neurons, which is $O(W_i \cdot W_{i+1})$. For the entire forward propagation process, the computational complexity of the agent is $O(S \cdot A \cdot \sum_{i=1}^{M-1} W_i \cdot W_{i+1})$, since the input and output layers are S and A, respectively.

In the routing process, a single demand is transmitted from the source node to the destination node hop-by-hop. Assume that, on average, a demand requires h hops to reach the destination node. After h hops, the computational complexity of agents in the routing process is $O(h \cdot S \cdot A \cdot \sum_{i=1}^{M-1} W_i \cdot W_{i+1}).$ Since there exist n demands and the routing process of each demand is independent, the total computational complexity of the MADDQN algorithm is $O(n \cdot h \cdot S \cdot A \cdot \sum_{i=1}^{M-1} W_i \cdot W_{i+1}).$

V. SIMULATION RESULTS

In this section, we conduct a couple of simulations utilizing python. The specific parameters are listed in Table I [34], [35],

[39]. In particular, UAVs are distributed in the area within a $1.5 \mathrm{km} \times 1.5 \mathrm{km}$ range, and the flight altitude range is within [0.12, 0.14]km. The minimum safe distance between UAVs is $d_{min} = 0.01 \mathrm{km}$. The number of demands and the trust value of all UAVs are initialized randomly. Besides, the size of each demand is randomly set within [400, 600]kbits. $E_{\rm elec}$ and ξ_{fs} are set as 1.5×10^{-4} J/bit and 2.5×10^{-8} J/(bit×m²), respectively [37].

To evaluate the designed adaptive weight methods, two baseline schemes are compared and detailed as follows.

• Average weight methods: The method indicates that $\psi^1_{i(t)}$ and $\psi^2_{i(t)}$ are the average of $1-\psi^0_{i(t)}$, while the calculation of $\psi^0_{i(t)}$ is the same as (36), i.e.,

$$\begin{cases} \psi_{i(t)}^{0} = 0.5 \times \frac{\mathbb{T}_{thr}}{\mathbb{T}_{i(t)}}, \\ \psi_{i(t)}^{1} = \psi_{i(t)}^{2} = \frac{1 - \psi_{i(t)}^{0}}{2}. \end{cases}$$
 (50)

• Random weight methods: The random weight method indicates that $\psi^1_{i(t)}$ is a uniformly distributed random variable over the interval $[0.2(1-\psi^0_{i(t)}), 0.8(1-\psi^0_{i(t)})]$, i.e.,

$$\begin{cases} \psi_{i(t)}^{0} = 0.5 \times \frac{\mathbb{T}_{thr}}{\mathbb{T}_{i(t)}}, \\ \psi_{i(t)}^{1} \sim U(0.2(1 - \psi_{i(t)}^{0}), 0.8(1 - \psi_{i(t)}^{0})), \\ \psi_{i(t)}^{2} = 1 - \psi_{i(t)}^{0} - \psi_{i(t)}^{1}. \end{cases}$$
(51)

Besides, to simulate the efficiency of the proposed adaptive weight method, the experimental configurations are as follows.

- The scale of UAV network is N = 20, and the number of malicious nodes is f = 2.
- The initial trust value is set as 1, which represents that the UAVs are completely trustworthy when departing from an initial station on the ground. Additionally, considering the high security requirements, the trust threshold is set as 0.8 [45].
- According to [45], the probability of malicious behaviors is set within [0.1, 0.5]. Therefore, in the paper, the demand delivery probability and correct path selection probability of malicious UAVs are characterized by p₁ ∈ {0.5, 0.7, 0.9} and p₂ ∈ {0.5, 0.7, 0.9}, respectively. Further, the simulation parameter space P of (p₁, p₂) is

$$\mathbb{P} = \begin{bmatrix} (0.5, 0.5) & (0.5, 0.7) & (0.5, 0.9) \\ (0.7, 0.5) & (0.7, 0.7) & (0.7, 0.9) \\ (0.9, 0.5) & (0.9, 0.7) & (0.9, 0.9) \end{bmatrix}. \tag{52}$$

• The evaluation metric defined as the minimum time step that can identify all malicious UAVs under various simulation parameter (p_1, p_2) in \mathbb{P} , i.e.,

$$\Upsilon_i^{\text{detect}} = \min\{t \in \mathbb{N}^+ | \mathbb{T}_{i(t)} < \mathbb{T}_{thr}\}, \forall i \in \mathcal{U}, t \in T.$$
(53)

In Fig. 3, the proposed adaptive weight method calculating the trust value is compared with the average and random weight methods. Specifically, we assess the metric of the minimum

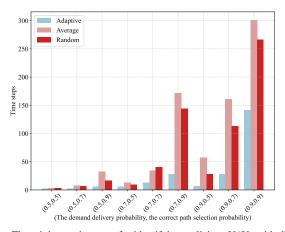


Fig. 3. The minimum time step for identifying malicious UAVs with different simulation parameters in space \mathbb{P} .

time steps required to identify all malicious UAVs, across a diverse range of simulation parameters in (52). It can be observed that when $(p_1, p_2) = (0.5, 0.5)$, it needs the fewest time steps for identifying the malicious UAVs. In contrast, when $(p_1, p_2) = (0.9, 0.9)$, the most time steps are required. The reason lies that the decreasing values of p_1 and p_2 indicate that the incorrect behaviors of malicious UAVs occur more frequently, leading to the increment of the evaluated incorrect behavior rates (i.e., $1 - \mathbb{T}^{dr}_{i(t)}$ and $1 - \mathbb{T}^{tp}_{i(t)}$ in formula (36)). Besides, it is noted that the adaptive method consistently obtains the minimum time steps compared with the other methods in various simulation parameters. This is due to the fact that the trust weights in the proposed method are proportional to incorrect behavior rates, and thus the trust values of UAVs are dynamically adjusted at a quadratic rate. The quadratic variation can impose more severe penalties on incorrect behaviors and accelerate the detection of malicious UAVs. Therefore, under the same case, the proposed adaptive weight method performs better than both the average and random weight methods.

In Fig. 4, we provide accumulative rewards to evaluate the convergence performance of the proposed BTMM-MADDQN algorithm under different learning rates, UAV numbers, and demand numbers, respectively. Besides, the number of malicious UAVs is 2. In Fig. 4(a), for all learning rates, the number of UAVs is 20 and the number of demands is set as 25. It can be observed that different learning rates have different performances on the convergence. In the first 2,000 episodes, rewards are not satisfactory. As the number of episodes increases, rewards increase and converge at the specific values. In detail, the reward at the learning rate of 0.0001 outperforms the rewards of other learning rates. Besides, when the value is 0.0005, the convergence performance is the worst and does not converge to a relatively stable result, which illustrates that a large learning rate may lead to a local solution. Considering the execution effect analyzed above, a learning rate of 0.0001 is selected for the rest of simulations.

In the network with 20 UAVs, Fig. 4(b) shows the rewards of the BTMM-MADDQN algorithm with different demand numbers. It is noted that the case of 40 demands obtains the

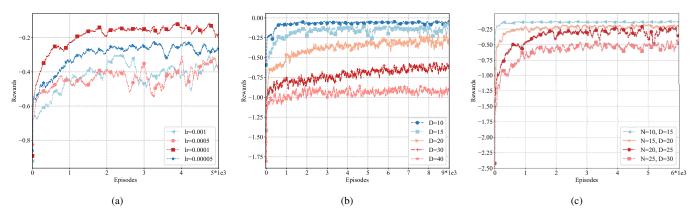


Fig. 4. Comparisons of the accumulative reward of episodes obtained by total UAVs for transmitting the given demands during the training process with 2 malicious UAVs. (a) Different learning rates with the same 20 UAVs and 25 demands. (b) Different demand numbers with the same 10 UAVs. (c) Different UAV numbers with different demand numbers.

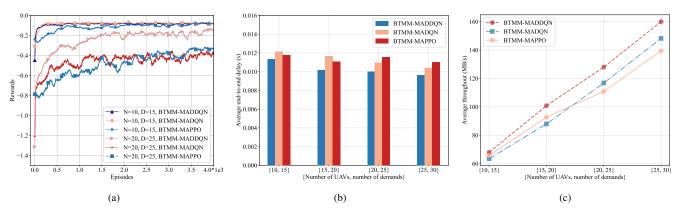


Fig. 5. Comparisons of the three algorithms in terms of the convergence performance, delay cost, and throughput with 2 malicious UAVs, respectively. (a) Rewards of episodes with different scales of UAV networks. (b) Average end-to-end delay with different UAV and demand numbers. (c) Average throughput with different UAV and demand numbers.

slowest convergence rate and the smallest reward. Meanwhile, as the number of demands decreases, the convergence rate is faster and the reward is larger. Since when there exist more demands, the environment perceived by UAVs is more complex and they require more episodes to learn, leading to a slower convergence speed. Besides, as the reward designed in (41) is related to the negative value of the summed delay for transmitting all demands, the more demands obtain the lower rewards.

In Fig. 4(c), the reward is evaluated in different scales of UAV networks with different demand numbers. Although the increased number of UAVs provides more resources to serve, the reward value decreases as the number of demands increases, since the reward is the sum of delay generated by all UAVs transmitting all demands. Besides, the convergence is slower with the scale of UAV networks increases and the demand number grows, since each agent needs more episodes to learn the more complex information of neighbor UAVs. In short, Fig. 4 illustrates that the BTMM-MADDQN algorithm effectively finds routing paths via maximizing convergent reward values, under various learning rates, UAV network scales, and demand numbers.

In Fig. 5, the BTMM-MADDQN is compared with BTMM-

MAPPO and BTMM-MADQN algorithms with regard to the convergence performance, average end-to-end delay, and average throughput. Specifically, in Fig. 5(a), the rewards of all three algorithms converge. When the numbers of UAVs and demands are same, for BTMM-MADDQN, the convergence speed is faster and rewards are larger compared to the BTMM-MAPPO and BTMM-MADQN algorithms. Specially, in networks with 10 UAVs or 20 UAVs, the effects of BTMM-MAPPO are the worst among the three algorithms. The end-toend delay for transmitting demands is evaluated in Fig. 5(b). It is observed that as the network becomes larger and demands grow, the delay of data transmission increases in all three algorithms. Nevertheless, the delay of the proposed BTMM-MADDQN algorithm decreases by 13.39% and 12.74% than the BTMM-MAPPO algorithm at {20,25} and the BTMM-MADQN algorithm at $\{15, 20\}$, respectively. Fig. 5(c) shows the performance of throughput in networks with different UAV and demand numbers. As the number of UAVs increases, the communication channel resources of networks are richer, and thus more demands are transmitted per second, leading to an increment in throughput. When the UAV number is 10, the network has the lowest throughput of 67.94MB/s for the proposed algorithm. Besides, the throughput of the BTMM-

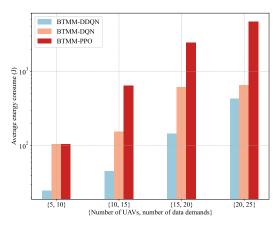


Fig. 6. The energy consumption of UAVs with different UAV numbers and demand numbers under different algorithms.

MADDQN algorithm increases by 15.44% and 14.6% than the BTMM-MAPPO algorithms at $\{20, 25\}$ and BTMM-MADQN algorithms at $\{15, 20\}$, respectively.

Fig. 6 illustrates the average energy consumption of UAVs with different scales of networks and demands. It is noted that as the number of demands increases, it requires more resources to complete the requirement, leading to the increased energy consumption. However, the energy consumption of the proposed BTMM-MADDQN is less than other algorithms, due to the less delay required by the demands. In detail, when the transmission delay is shorter, demands can reach the destination UAVs with the greater speed and precision, reducing the energy consumption caused by possible suboptimal routing. Besides, it is beneficial for each UAV to obtain the information of other UAVs timely, strengthening the collaborative operation among UAVs, and thus avoiding the unnecessary energy waste.

With different UAV network scales, Fig. 7 verifies the superiority of the proposed BTMM-MADDQN algorithm in terms of delay when there exist malicious UAVs. Specifically, in the UAV network without malicious nodes, the MADDQN-based routing method shows the excellent low delay performance, since UAVs do not generate malicious behaviors such as transmitting data via error paths with extra delay. However, in UAV networks with malicious nodes, the MADDQN-based approach obtains the worst performance of delay, since malicious UAVs may transmit demands under non-optimum paths, leading to the increased delay cost. Meanwhile, we can observe that the delay of the BTMM-MADDQN-based algorithm decreases by 16.6% than MADDQN-based approaches when there exist malicious UAVs.

Fig. 8 illustrates the performance of delay when there exist different numbers of malicious UAVs in different UAV network scales. It can be seen that the delay is the lowest in networks without malicious UAVs, while there exist malicious UAVs, the delay increases significantly. It indicates that malicious UAVs in routing paths reduce the delay performance. Besides, as the number of malicious UAVs increases, the delay grows sharply. It is accounted by the fact that the original routing paths are corrupted and replaced by suboptimal routing paths.

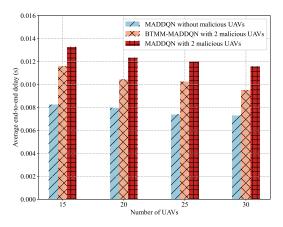


Fig. 7. Average end-to-end delay with different UAV numbers, under different algorithms and malicious UAV numbers.

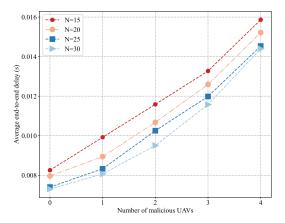
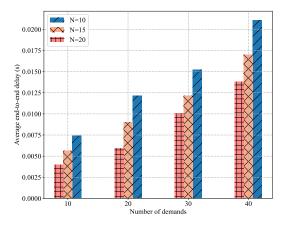


Fig. 8. Average end-to-end delay of the proposed BTMM-MADDQN with different network scales and malicious UAV numbers.

Meanwhile, as the number of UAVs becomes larger, the total network resources are richer in terms of the energy, channel, and queue capacity, resulting in the lower delay cost of routing.

In Fig. 9, the impacts of demands and UAVs on the routing performance are evaluated. Specifically, Fig. 9(a) depicts the delay cost of different demand numbers with different numbers of UAVs. As the demand number increases, the total endto-end delay rises, due to the increased congestion in UAV networks. Nevertheless, with the same demands, the delay decreases as the UAV number increases, since the network service is more resourced and the demand is transmitted at a high probability without queuing on UAVs. Fig. 9(b) depicts the average queue length on UAVs for different demands in networks. It is noted that as the number of demands increases, the queue length rises quickly, due to the increased congestion in UAV networks. Meanwhile, for the same amount of the demand to be transmitted in networks, the average queue length is smaller when UAV numbers are larger, due to the more adequate service resources. Besides, when the demand numbers are greater than UAV numbers, the average queue length on each UAV increases at a greater speed.



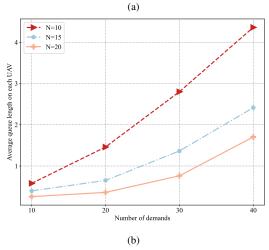


Fig. 9. Comparisons of different scales of UAV networks with 2 malicious UAVs. (a) Average end-to-end delay cost with different demand numbers. (b) The average queue length on each UAVs with different demand numbers.

VI. CONCLUSIONS

In this work, we depict the routing process in a timevarying UAV network with malicious nodes. Besides, the routing problem is formulated to minimize the total end-toend delay with multi-constraints. To enhance the safety during routing processes, we put forward the BTMM, in which the node trust evaluation mechanism is designed, and consensus UAV update mechanisms are proposed to improve the security of PBFT algorithms. Further, to deal with the challenge of obtaining global information in decentralized UAV networks, we reformulate the routing problem into a Dec-POMDP, and an MADDQN-based algorithm is proposed to solve it. Simulation results in UAV networks with attacked nodes demonstrate that the designed BTMM-MADDQN algorithm outperforms in multi aspects, such as the convergence performance, delay, throughput, and queue length, and in particular the proposed algorithm decreases delay by 16.6\% than methods without the BTMM.

REFERENCES

[1] B. He, X. Ji, G. Li, and B. Cheng, "Key technologies and applications of UAVs in underground space: A review," *IEEE Trans. Cognit. Commun. Networking*, vol. 10, no. 3, pp. 1026–1049, Jan. 2024.

- [2] Z. Jia, Q. Wu, C. Dong, C. Yuen, and Z. Han, "Hierarchical aerial computing for Internet of things via cooperation of HAPs and UAVs," *IEEE Internet Things J.*, vol. 10, no. 7, pp. 5676–5688, Apr. 2023.
- [3] S. Javed, A. Hassan, R. Ahmad, W. Ahmed, R. Ahmed, A. Saadat, and M. Guizani, "State-of-the-art and future research challenges in UAV swarms," *IEEE Internet Things J.*, vol. 11, no. 11, pp. 19023–19045, Jun. 2024.
- [4] D. Shumeye Lakew, U. Sa'ad, N.-N. Dao, W. Na, and S. Cho, "Routing in flying ad hoc networks: A comprehensive survey," *IEEE Commun. Surv. Tutorials*, vol. 22, no. 2, pp. 1071–1120, 2nd Quart. 2020.
- [5] Y. Ke, K. Huang, X. Qiu, B. Song, L. Xu, J. Yin, and Y. Yang, "Distributed routing optimization algorithm for FANET based on multiagent reinforcement learning," *IEEE Sens. J.*, early access, Jun. 2024.
- [6] Z. Lu, Z. Jia, Q. Wu, and Z. Han, "Joint trajectory planning and communication design for multiple UAVs in intelligent collaborative airground communication systems," *IEEE Internet Things J.*, early access, Jun. 2024.
- [7] W. Z. Khan, Q.-u.-A. Arshad, S. Hakak, M. K. Khan, and Saeed-Ur-Rehman, "Trust management in social Internet of things: Architectures, recent advancements, and future challenges," *IEEE Internet of Things J.*, vol. 8, no. 10, pp. 7768–7788, May 2021.
- [8] D. E. Kouicem, Y. Imine, A. Bouabdallah, and H. Lakhlef, "Decentralized blockchain-based trust management protocol for the Internet of things," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 2, pp. 1292–1306, Mar./Apr. 2022.
- [9] J. Leng, M. Zhou, J. L. Zhao, Y. Huang, and Y. Bian, "Blockchain security: A survey of techniques and research directions," *IEEE Trans. Serv. Comput.*, vol. 15, no. 4, pp. 2490–2510, Jul./Aug. 2022.
- [10] Y. Wang, Z. Su, Q. Xu, R. Li, T. H. Luan, and P. Wang, "A secure and intelligent data sharing scheme for UAV-assisted disaster rescue," *IEEE/ACM Trans. Networking*, vol. 31, no. 6, pp. 2422–2438, Dec. 2023.
- [11] R. Pramudita, H. Heryanto, R. Trias Handayanto, D. Setiyadi, R. W. Arifin, and N. Safitri, "Shortest path calculation algorithms for geographic information systems," in *Proc. IEEE Int. Conf. Inform. Comput.*, Semarang, Indonesia, Feb. 2019.
- [12] Y. Cui, H. Tian, C. Chen, W. Ni, H. Wu, and G. Nie, "New geographical routing protocol for three-dimensional flying ad-hoc network based on new effective transmission range," *IEEE Trans. Veh. Technol.*, vol. 72, no. 12, pp. 16135–16147, Dec. 2023.
- [13] Z. Jia, J. You, C. Dong, Q. Wu, F. Zhou, D. Niyato, and Z. Han, "Cooperative cognitive dynamic system in UAV swarms: Reconfigurable mechanism and framework," *IEEE Veh. Technol. Mag.*, early access, Jul. 2024.
- [14] C. Amato, G. Chowdhary, A. Geramifard, N. K. Üre, and M. J. Kochenderfer, "Decentralized control of partially observable Markov decision processes," in *Proc. 52nd IEEE Conf. Decis. Control*, Firenze, Italy, Dec. 2013, pp. 2398–2405.
- [15] V. Varadharajan, K. K. Karmakar, U. Tupakula, and M. Hitchens, "To-ward a trust aware network slice-based service provision in virtualized infrastructures," *IEEE Trans. Netw. Serv. Manage.*, vol. 19, no. 2, pp. 1065–1082, Jun. 2022.
- [16] Y. Yu, Q. Lu, and Y. Fu, "Dynamic trust management for the edge devices in industrial Internet," *IEEE Internet of Things J.*, vol. 11, no. 10, pp. 18410–18420, May 2024.
- [17] T. Khan, K. Singh, M. H. Hasan, K. Ahmad, G. T. Reddy, S. Mohan, and A. Ahmadian, "ETERS: A comprehensive energy aware trust-based efficient routing scheme for adversarial WSNs," *Future Gener. Comput. Syst.*, vol. 125, pp. 921–943, Dec. 2021.
- [18] I. U. Din, A. Bano, K. A. Awan, A. Almogren, A. Altameem, and M. Guizani, "Lighttrust: Lightweight trust management for edge devices in industrial Internet of things," *IEEE Internet of Things J.*, vol. 10, no. 4, pp. 2776–2783, Feb. 2023.
- [19] H. Lu, Y. Tang, and Y. Sun, "DRRS-BC: Decentralized routing registration system based on blockchain," *IEEE/CAA J. Automatica Sinica*, vol. 8, no. 12, pp. 1868–1876, Dec. 2021.
- [20] X. Jian, P. Leng, Y. Wang, M. Alrashoud, and M. S. Hossain, "Blockchainempowered trusted networking for unmanned aerial vehicles in the B5G era," *IEEE Netw.*, vol. 35, no. 1, pp. 72–77, Jan./Feb. 2021.
- [21] J. Yang, X. Liu, X. Jiang, Y. Zhang, S. Chen, and H. He, "Toward trusted unmanned aerial vehicle swarm networks: A blockchain-based approach," *IEEE Veh. Technol. Mag.*, vol. 18, no. 2, pp. 98–108, Jun. 2023.
- [22] Y. Zhang, K. Gai, J. Xiao, L. Zhu, and K.-K. R. Choo, "Blockchain-

- empowered efficient data sharing in Internet of things settings," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 12, pp. 3422–3436, Dec. 2022.
- [23] D. Chattaraj, B. Bera, A. K. Das, J. J. P. C. Rodrigues, and Y. Park, "Designing fine-grained access control for software-defined networks using private blockchain," *IEEE Internet Things J.*, vol. 9, no. 2, pp. 1542–1559, Jan. 2022.
- [24] R. Li, Y. Qin, C. Wang, M. Li, and X. Chu, "A blockchain-enabled framework for enhancing scalability and security in IIoT," *IEEE Trans. Ind. Inf.*, vol. 19, no. 6, pp. 7389–7400, Jun. 2023.
- [25] Z. Han, C. Xu, S. Ma, Y. Hu, G. Zhao, and S. Yu, "DTE-RR: Dynamic topology evolution-based reliable routing in VANET," *IEEE Wireless Commun. Lett.*, vol. 12, no. 6, pp. 1061–1065, Jun. 2023.
- [26] L. Zhu, C. Li, B. Li, X. Wang, and G. Mao, "Geographic routing in multilevel scenarios of vehicular ad hoc networks," *IEEE Trans. Veh. Technol.*, vol. 65, no. 9, pp. 7740–7753, Sep. 2016.
- [27] M. Z. Hasan and F. Al-Turjman, "Optimizing multipath routing with guaranteed fault tolerance in Internet of things," *IEEE Sens. J.*, vol. 17, no. 19, pp. 6463–6473, Oct. 2017.
- [28] M. Y. Arafat and S. Moh, "A Q-learning-based topology-aware routing protocol for flying ad hoc networks," *IEEE Internet Things J.*, vol. 9, no. 3, pp. 1985–2000, Feb. 2022.
- [29] A. A. Okine, N. Adam, F. Naeem, and G. Kaddoum, "Multi-agent deep reinforcement learning for packet routing in tactical mobile sensor networks," *IEEE Trans. Netw. and Serv. Manage.*, vol. 21, no. 2, pp. 2155–2169, Apr. 2024.
- [30] Y. Peng, Y. Liu, D. Li, and H. Zhang, "Deep reinforcement learning based freshness-aware path planning for UAV-assisted edge computing networks with device mobility," *Remote Sens.*, vol. 14, no. 16, pp. 4016–4035, Aug. 2022
- [31] R. Ding, J. Chen, W. Wu, J. Liu, F. Gao, and X. Shen, "Packet routing in dynamic multi-hop UAV relay network: A multi-agent learning approach," *IEEE Trans. Veh. Technol.*, vol. 71, no. 9, pp. 10059–10072, Sep. 2022.
- [32] L. Chen, B. Hu, Z.-H. Guan, L. Zhao, and X. Shen, "Multiagent metareinforcement learning for adaptive multipath routing optimization," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 33, no. 10, pp. 5374–5386, Oct. 2022.
- [33] X. Huang, J. Chen, M. Cai, W. Wang, and X. Hu, "Traffic node importance evaluation based on clustering in represented transportation networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 9, pp. 16622–16631, Sep. 2022
- [34] S. He, Z. Jia, C. Dong, W. Wang, Y. Cao, Y. Yang, and Q. Wu, "Routing recovery for UAV networks with deliberate attacks: A reinforcement learning based approach," in *Proc. IEEE Glob. Commun. Conf.*, Kuala Lumpur, Malaysia, Dec. 2023, pp. 952–957.
- [35] Y. Chen, N. Zhao, Z. Ding, and M.-S. Alouini, "Multiple UAVs as relays: Multi-hop single link versus multiple dual-hop links," *IEEE Trans. Wireless Commun.*, vol. 17, no. 9, pp. 6348–6359, Sep. 2018.
- [36] A. Wyner, "Recent results in the shannon theory," *IEEE Trans. Inf. Theory*, vol. 20, no. 1, pp. 2–10, Jan. 1974.
- [37] Z. Wang, H. Yao, T. Mai, Z. Xiong, X. Wu, D. Wu, and S. Guo, "Learning to routing in UAV swarm network: A multi-agent reinforcement learning approach," *IEEE Trans. Veh. Technol.*, vol. 72, no. 5, pp. 6611–6624, May 2023.
- [38] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Trans. Wireless Commun.*, vol. 1, no. 4, pp. 660–670, Oct. 2002.
- [39] S. Zhou, Y. Cheng, X. Lei, Q. Peng, J. Wang, and S. Li, "Resource allocation in UAV-assisted networks: A clustering-aided reinforcement learning approach," *IEEE Trans. Veh. Technol.*, vol. 71, no. 11, pp. 12088– 12103, Nov. 2022.
- [40] L. Zhang, Z. Mu, and X. Gao, "Coupling analysis and performance study of commercial 18650 lithium-ion batteries under conditions of temperature and vibration," *Energies*, vol. 11, no. 10, pp. 2856–2882, Oct. 2018.
- [41] M. Mizat, B. Bendjedia, and N. Rizoug, "Influence of battery technologies on UAVs characteristics," in *Proc. 1st Int. Conf. Renew. Solutions Ecosyst. Towards Sustain. Energy Transition (ICRSEtoSET)*, Kuala Lumpur, Malaysia, May 2023.
- [42] G. Lancia and P. Serafini, Compact Extended Linear Programming Models. Cham, Switzerland: Springer, 2018.
- [43] J. Jiang, G. Han, F. Wang, L. Shu, and M. Guizani, "An efficient distributed trust model for wireless sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 5, pp. 1228–1237, May 2015.

- [44] H. Fatemidokht, M. K. Rafsanjani, B. B. Gupta, and C.-H. Hsu, "Efficient and secure routing protocol based on artificial intelligence algorithms with UAV-assisted for vehicular ad hoc networks in intelligent transportation systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 7, pp. 4757–4769, Jul. 2021.
- [45] Z. Teng, B. Pang, M. Sun, L. Xie, and L. Guo, "A malicious node identification strategy with environmental parameters optimization in wireless sensor network," Wireless Pers. Commun., vol. 117, pp. 1143– 1162, Mar. 2021.
- [46] L. Liu, J. Feng, C. Wu, C. Chen, and Q. Pei, "Reputation management for consensus mechanism in vehicular edge metaverse," *IEEE J. Sel. Areas Commun.*, vol. 42, no. 4, pp. 919–932, Apr. 2024.
- [47] X. Geng and B. Zhang, "Deep Q-network-based intelligent routing protocol for underwater acoustic sensor network," *IEEE Sens. J.*, vol. 23, no. 4, pp. 3936–3943, Feb. 2023.
- [48] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proc. 13th AAAI Conf. Artif. Intell.*, vol. 30, no. 1, Phoenix, Arizona, Feb. 2016, pp. 2094–2100.