On the Joint Placement of Blockchain and Users' Virtualized Services in the Internet of Vehicles

Messaoud Ait Yahia*, Mouhamadou Mouctar Gueye*, Wael Jaafar*, Rami Langar*‡
* Software and IT Engineering Department, Ecole de Technologie Supérieure (ÉTS), Montréal, QC H3C 1K3, Canada
‡ LIGM-CNRS UMR 8049, University Gustave Eiffel, F-77420 Marne-la-Vallée, France
E-mails:{messaoud.ait-yahia.1, mouhamdou-mouctar.gueye.1}@ens.etsmtl.ca; {wael.jaafar, rami.langar}@etsmtl.ca

***Abstract*—Driven by the evolution of the Internet of Things (IoT), Intelligent Transportation Systems are rapidly developing from traditional vehicle ad-hoc networks to the Internet of Vehicles (IoV). It supports various future applications, such as road traffic status detection and anti-collision warning for autonomous driving. IoV networks face several security and privacy challenges. Indeed, malicious nodes can compromise the system's integrity and confidentiality by sending fake messages, thus jeopardizing human life. Blockchain has emerged as a potential solution to enhance IoV's security given its unique features. To do so, this paper proposes a novel approach for joint resource allocation of blockchain and users' virtualized services in IoV networks. Specifically, we formulate the joint placement problem of blockchain and user services' virtualized network functions (VNFs) aiming to maximize the satisfaction of users' service requests while realizing the shortest blockchain operation times related to those requests, as an integer linear program (ILP), known to be NP-hard. To solve it in a reasonable time, we propose two low-complexity solutions, the first is a meta-heuristic based on particle swarm optimization (PSO) and the second is a greedy solution. Through simulations, we evaluate the effectiveness of these approaches and their suitability for different service requirements.**

***Index Terms*—Blockchain, IoV, VNF Placement, PSO, access control, edge computing, wireless network.**

I. Introduction

The Internet of Vehicles (IoV) has become an integral part of Intelligent Transport Systems (ITS), enabling various future applications such as autonomous driving, collision warning, road traffic condition detection, and vehicle service data sharing to improve the overall traveling experience [1]. However, ensuring the security and privacy of IoV systems is highly challenging due to the mobility and high-speed nature of vehicles. Indeed, since IoV communication links are over the air, malicious and unauthorized nodes can easily invade IoV systems to compromise data integrity, availability, and confidentiality. These cyber-attacks may include replay and camouflage attacks, or message tampering attacks [2]. Relying on blockchain to secure distributed systems such as IoV has attracted interest in recent years. Given blockchain's immutability, decentralization, and traceability, it can for instance guarantee vehicle access control by verifying the integrity and authenticity of virtualized service requests using smart contracts and asymmetric encryption [3]. However, in latency-sensitive applications such as IoV, user services need to be provided at the edge of the network to satisfy the stringent quality-of-service (QoS) of ITS services. Consequently, accurate and optimized resource allocation for virtualized network functions (VNFs) at the edge of blockchain-enabled IoV becomes a significant challenge.

This work was supported in part by the ANR 5G-INSIGHT project (Grant no. ANR-20-CE25-0015), and in part by the Innovation for Defence Excellence and Security (IDEaS) program of the Department of National Defence Canada.

To the best of our knowledge, this work is among the firsts to investigate resource allocation for virtualized blockchain-enabled IoV where the placement of blockchain and user services' VNFs is jointly optimized. Hence, we aim in this work to design an efficient blockchain-based IoV framework for improved security while satisfying user requests' QoS. The main contributions of this paper can be summarized as follows:

1) We propose a novel blockchain-enabled framework tailored for virtualized IoV networks.
2) We formulate the joint blockchain and user services VNF placement problem as an ILP, aiming to maximize the acceptance/satisfaction rate of users' service requests, while achieving the minimum delay for blockchain operations to validate those requests.
3) Given the high latency of the ILP-based solution, we propose two approaches to solve the formulated problem with low time complexity, namely a PSO-based and a greedy-based algorithm.
4) Through simulations, we demonstrate the efficiency of the proposed solutions and their suitability for blockchain-enabled IoV networks, in terms of blockchain validation delays and acceptance ratio of users' service requests.

The remaining of the paper is organized as follows. Section II presents the related works. Section III describes the system model and Section IV formulates the related problem. Section V exposes the proposed VNF placement solutions, while Section VI illustrates the simulation results. Finally, Section VII closes the paper.

II. Related Work

Previous research works explored the use of blockchain to improve the security of IoV networks. For instance, authors of [2] proposed an intelligent access control mechanism based on blockchain to protect against malicious and unauthorized nodes that can break the integrity, availability, and confidentiality of IoV systems. Specifically, they predicted the risk level based on the historical behaviors of the requesting vehicles with access rules and policies stored in the blockchain as smart contracts. In [4], the authors integrated the blockchain to guarantee the integrity of mutual authentication results among vehicles, edge nodes, and cloud servers, as well as to prevent impersonation attacks from both edge nodes and vehicles. In [1], the authors proposed a decentralized framework for multi-party participation to defend against malicious attacks such as reporting fake information and selfish behavior. They designed smart contracts to securely manage the reputation service for vehicles.

However, previous studies did not address the challenges associated with blockchain for latency-sensitive applications. To do so, authors of [5] proposed blockchain to improve the security of task offloading in a mobile edge computing (MEC) system. They formulated the problem of joint task offloading and resource allocation to minimize time and energy consumption to complete tasks and solved it using a deep Q-learning-based algorithm. Their approach did not account for the delay caused by the blockchain's operation. In [6], a novel blockchain-based access control mechanism for interactions between vehicles and roadside units (RSUs) is introduced. Given the high latency of public blockchain hosted on the cloud, the authors opted for a lightweight version that somewhat reduced security. Finally, in [7], the authors used blockchain to guarantee the authenticity of data transmission in IoV networks. They formulated the related problem of blockchain transactional throughput maximization and energy consumption and computation overhead minimization. To solve it, they developed an asynchronous advantage actor-critic (A3C)-based algorithm that optimizes several parameters such as task offloading strategy, caching policy, number of offloaded consensus nodes, block interval, and block size. However, offloading blockchain tasks to the cloud raises concerns about privacy, data leakage, and increased latency. In addition, while a larger block size can improve blockchain transactions throughput, it can also lead to bandwidth overload, which is not addressed in this study.

III. System Model

This section presents the blockchain-enabled IoV framework. We start by presenting the consortium blockchain and its workflow, followed by the communication and computation models, and then by the delay analysis. The key symbols used in the remaining of the section are summarized in Table I.

We consider a wireless IoV network as shown in Fig. 1, where an unmanned aerial vehicle (UAV) located at altitude H acts as a relay to handle communications between N gNodeBs (gNBs) of a set $\mathcal{N}$ where $N = |\mathcal{N}|$. We use UAV as a relay to provide opportunistic line-of-sight links and further assistance to the terrestrial communications [8]. The UAV's Cartesian coordinates are given by $\mathbf{q}_u = [x_u, y_u, H]$. Similarly, the location of a gNB i is given by $\mathbf{q}_i = [x_i, y_i, H_i]$, where H_i represents its altitude level such that $H_i < H$. Hence, the distance between the UAV and gNB i is given by $L_{u,i} = \|\mathbf{q}_u - \mathbf{q}_i\| = \sqrt{(x_u - x_i)^2 + (y_u - y_i)^2 + (H - H_i)^2} = \sqrt{(H - H_i)^2 + ||\mathbf{l}_u - \mathbf{l}_i||^2}$, where $\mathbf{l}_u$ and $\mathbf{l}_i$ are the horizontal coordinates of the UAV and gNB i, respectively.

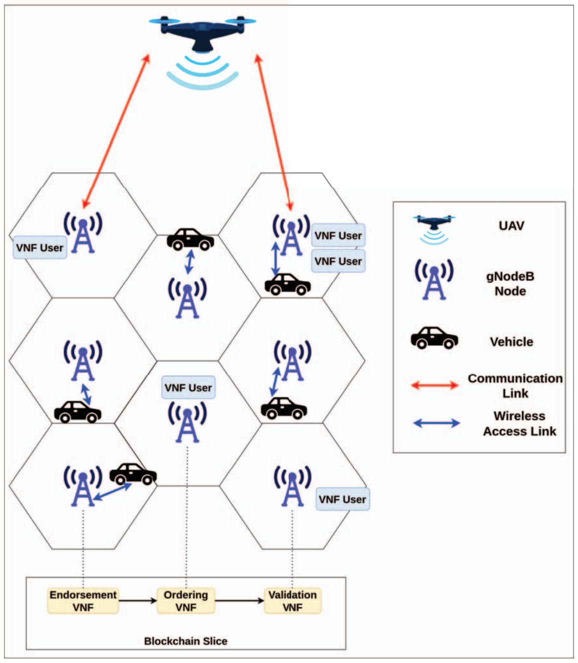

Fig. 1. System Model.

A. Consortium Blockchain

In this work, we use the private Hyperledger Fabric network with the RAFT consensus mechanism [9] as our blockchain network. Hyperledger Fabric is an open-source permissioned blockchain platform established under the Linux Foundation [10]. It introduces a new architecture for transaction flow, different from Bitcoin [11] and Ethereum [12] networks, called execute-order-validate. The transaction flow is separated into three steps, as shown in Fig. 2. In the first phase, a.k.a., the endorsement phase, the correctness of the transaction is checked by executing the smart contract without making changes to the blockchain ledger. This step is crucial since it eliminates any non-determinism and filters out any inconsistency before creating the block. The second phase is the ordering one. In which a consensus protocol, i.e., RAFT [9], is executed to order the transactions and create a new block. Finally, during the last phase, called the validation phase, the block is validated against an endorsement policy before being committed to the ledger. Note that blocks failing to pass validation are discarded.

For our system, several types of nodes are involved: Client nodes, which are the gNBs, orderer nodes (leaders and followers), and validator nodes (peer nodes). Validator nodes are further divided into two categories: Endorsers who participate in both the endorsement and validation phases, and non-endorsers (committing peers), which are only involved in the validation process. In a virtualized network, these node types are transposed into specific instances of virtualized network functions (VNFs) that can be deployed on gNBs and form a

TABLE I
KEY SYMBOLS' DESCRIPTION

Symbol	Description
$\mathcal{N}$	A set of gNodeBs (gNBs).
$\mathcal{M}$	A set of vehicle users.
$\mathcal{N}_{\text{VNF}}$	A set of requested VNFs by vehicle users.
$\mathcal{S}$	A set of blockchain nodes. $\mathcal{S} = \{\mathcal{S}_e, \mathcal{S}_{ol}, \mathcal{S}_{of}, \mathcal{S}_v\}$ are endorser, orderer leader, orderer follower, and validator nodes, respectively.
n_e, n_{ol}, n_{of}, n_v	The number of endorser, orderer (leaders and followers), validator nodes respectively.
R_i^{CPU}, R_i^{stor}	Computing power and storage capacity of gNB i.
I_x^{CPU}	With $x \in \{e, ol, of, v\}$, are the CPU cycles required to complete endorsement, ordering processing (For the leader and followers), and validation of transactions or blocks, respective.
$I_{\text{UE,j}}^{\text{CPU}}$	The number of CPU cycles needed to satisfy the request of user j (in cycles per task).
$\beta_k \in [0, 1]$	Occupation ratio of the computing capacity of a gNB to handle blockchain VNF type k.
G	The number of blockchain blocks that must be guaranteed.
C_b	The size of one block in the blockchain network.
$\mathbf{q}_u$, $\mathbf{q}_i$	The location of the UAV u and a gNB i respectively.
H, H_i	The altitude level of the UAV u and a gNB i.
$L_{u,i}$	The distance between the UAV u and a gNB i
$W_{u,x}$	With $x \in \{e, ol, of, v\}$, is the size of data transmitted from the UAV u to the blockchain nodes.
$R_{u,i}$	Transmission rate between the UAV u and a gNB i.
$\sigma_{u,i}$	The unitary Additive White Gaussian noise (AWGN) power.
P_i, P_u	The transmit power of the gNB i and UAV u ,respectively.
B	The allocated bandwidth for uplink/downlink communication.
$G_{u,i}$	The channel gain between the UAV u and a gNB i.
g_0	The channel gain at reference distance 1 m.
ζ	The attenuation factor under non-line-of-sight (NLoS).
$\beta_{u,i}$	Elevation angle between the UAV u and a gNB i.
a , b	The environment-related parameters.
$\epsilon_{u,i}$	The probability of line-of-sight (LoS).
$D_{u,x}^{\text{comm}}$	With $x \in \{e, ol, of, v\}$, is the transmission delay between the UAV u and the blockchain nodes.
$D_{e,j}^{\text{comp}}, D_{ol,j}^{\text{comp}}, D_{of,j}^{\text{comp}}, D_{v,j}^{\text{comp}}$	The computing delay for executing the endorsement, ordering, and validation processes, respectively, related to the VNF request of user j.
D_j	The overall blockchain validation delay for the VNF request j.
$D_{e,j}, D_{o,j}, D_{v,j}$	The endorsement, ordering, and validation delays related to the VNF request of user j.

dedicated 5G network slice, called *Blockchain Slice*. This slice consists of four VNF types, namely VNF$_e$, VNF$_{ol}$, VNF$_{of}$, and VNF$_v$ for virtual endorser, orderer-leader, virtual orderer-follower and validator, respectively. Hence, when a blockchain node is called by its role (e.g., endorser), it means that this node hosts a VNF$_e$ instance.

B. Communication Model

We assume that the wireless communication between the single-antenna UAV and single-antenna gNB i is characterized by a line-of-sight (LoS) probabilistic channel model, where the probability of LoS is expressed by

$$\epsilon_{u,i} = \frac{1}{1 + a\exp(-b(\beta_{u,i} - a))}, \; \forall i = 1, \ldots, N, \quad (1)$$

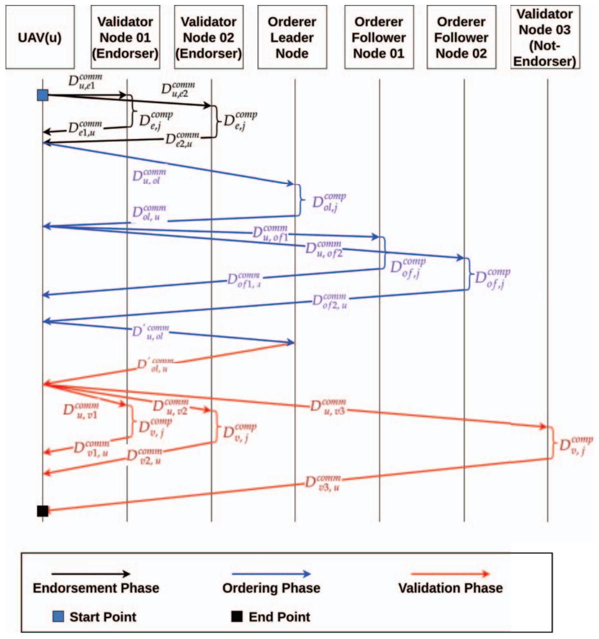

Fig. 2. The blockchain's workflow.

where a and b are parameters related to the environment and $\beta_{u,i} = \arctan\left(\frac{H - H_i}{\|\mathbf{l}_u - \mathbf{l}_i\|}\right)$ defines the elevation angle between the UAV and the gNB i. Consequently, the associated channel gain can be written as

$$G_{u,i} = g_0 \left(\frac{\epsilon_{u,i} + \zeta(1 - \epsilon_{u,i})}{(H - H_i)^2 + \|\mathbf{l}_u - \mathbf{l}_i\|^2} \right), \; \forall i = 1, \ldots, N, \quad (2)$$

where g_0 is the channel gain at reference distance 1 m and ζ reflects the attenuation factor under non-LoS (NLoS) conditions. We suppose that the uplink and downlink channels are symmetric. Hence, the associated achievable uplink/downlink data rate is given by:

$$R_{i,u} = B \log_2 \left(1 + \frac{P_i G_{u,i}}{B \sigma_{u,i}^2} \right), \; \forall i = 1, \ldots, N, \quad (3)$$

and

$$R_{u,i} = B \log_2 \left(1 + \frac{P_u G_{u,i}}{B \sigma_{u,i}^2} \right), \; \forall i = 1, \ldots, N, \quad (4)$$

where B is the allocated bandwidth for both uplink and downlink communication, P_i and P_u are the transmit power of the gNB i and UAV u, respectively, and $\sigma_{u,i}^2$ is the unitary Additive White Gaussian noise (AWGN) power. Thus, the transmission delays are expressed as

$$D_{u,i}^{\text{comm}} = \frac{W_{u,i}}{R_{u,i}}, \; \forall i = 1, \ldots, N, \quad (5)$$

and

$$D_{i,u}^{\text{comm}} = \frac{W_{i,u}}{R_{i,u}}, \; \forall i = 1, \ldots, N, \quad (6)$$

where $W_{u,i}$ (resp. $W_{i,u}$) represents the size of data transmitted from the UAV to gNB i (resp. from gNB i to UAV u).

C. Computing Model

Each gNB is equipped with a computing power R_i^{CPU} (in cycles per second) and a storage capacity R_i^{stor} (in Gigabytes).

Each vehicle j might require a service that needs to be executed as a user VNF instance deployed at the edge of the network. For the sake of simplicity, we assume here that users' service requests are modeled as individual VNF instances deployed at the gNBs. We denote by $\mathcal{N}_{\text{VNF}}$ the set of users' VNFs requested by $\mathcal{M}$ vehicles in the system, where $N_{\text{VNF}} = |\mathcal{N}_{\text{VNF}}|$. When a VNF (computing) demand is generated, it is associated with a smart contract and recorded within the blockchain ledger to guarantee traceability and enforce the security of handled users' VNFs requests.

We define by I_x^{CPU}, $x \in \{e, ol, of, v\}$ (in cycles per transaction or block) the CPU cycles required to complete endorsement, ordering processing for leaders and followers (i.e., to reach consensus), and validation of transactions or blocks, respectively. Similarly, the number of CPU cycles needed to satisfy a user VNF request j can be denoted by $I_{\text{UE,j}}^{\text{CPU}}$ (in cycles per task).

Since a gNB can host both user VNFs and blockchain VNFs, we define a parameter $\beta_k \in [0,1]$ as the occupation ratio of the computing capacity of a gNB to handle the blockchain service type k, such that $k \in \{1,2,3,4\}$ corresponds to the blockchain VNF type $\{\text{VNF}_e$, VNF_{ol}, VNF_{of}, and $\text{VNF}_v\}$, respectively. Note that $\beta_1 + \beta_4 \leq 1$ since an endorser node needs to deploy both VNF_e and VNF_v at the same time, as illustrated in Fig. 2.

D. Delay Analysis

We define D_j as the total blockchain validation delay for the user VNF request j. Since all communications to/from the UAV and IoV nodes are not dynamic (i.e., users are assumed to be static), D_j can be written as follows:

$$D_j = D_{e,j} + D_{o,j} + D_{v,j},\ j = 1, \ldots, N_{\text{VNF}}, \tag{7}$$

where $D_{e,j}$, $D_{o,j}$, and $D_{v,j}$ are the endorsement, ordering, and validation delays related to the VNF request of user j, respectively. We denote by $D_{u,x}^{\text{comm}}$, $x \in \{e, ol, of, v\}$ the communication delay between the UAV and endorser, order-leader, orderer-follower, and validator, respectively. Also, $D_{x,j}^{\text{comp}}$, $x \in \{e, ol, of, v\}$ denotes the computing delay to execute the related endorsement, ordering, and validation, respectively.

During endorsement, the UAV submits transactions related to VNF requests to the endorser nodes. The latter then simulate the smart contract without updating the ledger. Each endorser node sends the simulation results back to the UAV, which must wait for responses from all endorser nodes before proceeding to the next phase. Hence, the delay incurred during the endorsement process is calculated as

$$D_{e,j} = \max_{e \in \mathcal{S}_e} \left(D_{u,e}^{\text{comm}} + D_{e,j}^{\text{comp}} + D_{e,u}^{\text{comm}} \right), \quad \forall j = 1, \ldots, N_{VNF}, \tag{8}$$

where $\mathcal{S}_e$ is the set of endorsers. Following the endorsement phase, the UAV sends the transactions and endorsement results to the orderer nodes, who create a new block using the RAFT consensus mechanism. The total delay of the ordering phase includes the computing delay for the ordering process and the communication delay among the orderer nodes (orderer leader and orderer follower nodes). It is expressed by:

$$\begin{aligned} D_{o,j} = &\left(D_{u,ol}^{\text{comm}} + D_{ol,j}^{\text{comp}} + D_{ol,u}^{\text{comm}} \right) \\ &+ \max_{of \in \mathcal{S}_{of}} \left(D_{u,of}^{\text{comm}} + D_{of,j}^{\text{comp}} + D_{of,u}^{\text{comm}} \right) + D_{u,ol}^{'\text{comm}}, \\ &\forall j = 1, \ldots, N_{VNF}, \end{aligned} \tag{9}$$

where $\mathcal{S}_{of}$ is the set of orderer-follower nodes.

Subsequently, the block is broadcasted to all validator nodes for verification and commitment to the ledger. The time required to execute this phase is given by

$$D_{v,j} = D_{ol,u}^{'\text{comm}} + \max_{v \in \mathcal{S}_v} \left(D_{u,v}^{\text{comm}} + D_{v,j}^{\text{comp}} + D_{v,u}^{\text{comm}} \right), \quad \forall j = 1, \ldots, N_{VNF}, \tag{10}$$

where $\mathcal{S}_v$ is the set of validators such that $\mathcal{S}_v \cap \mathcal{S}_e = \mathcal{S}_e$.

IV. Problem Formulation

We define the binary matrix $\mathbf{V} = [V_{i,j}]$ of size $N \times N_{VNF}$ to reflect the binary decisions of allocating user VNFs to gNBs, i.e., $V_{i,j} = 1$ means that user VNF j is placed in gNB i, and where N_{VNF} is the number of VNF requests incoming. Also, let $\mathbf{x}_e = [x_{e,1}, x_{e,2}, \ldots, x_{e,N}]$, $\mathbf{x}_{ol} = [x_{ol,1}, x_{ol,2}, \ldots, x_{ol,N}]$, $\mathbf{x}_{of} = [x_{of,1}, x_{of,2}, \ldots, x_{of,N}]$, and $\mathbf{x}_v = [x_{v,1}, x_{v,2}, \ldots, x_{v,N}]$ be the binary decision vectors that determine the placement strategy of VNFs of type endorsement, ordering-leader, ordering-follower, and validation, into the gNBs, such that $\sum_{i=1}^{N} x_{e,i} = |\mathcal{S}_e|$, $\sum_{i=1}^{N} x_{ol,i} = 1$ (one leader only), $\sum_{i=1}^{N} x_{of,i} = |\mathcal{S}_{of}|$, and $\sum_{i=1}^{N} x_{v,i} = |\mathcal{S}_v|$. Hence, our problem can be formulated as follows:

$$\underset{\substack{\mathbf{V}, \mathbf{x}_e, \mathbf{x}_{ol}, \\ \mathbf{x}_{of}, \mathbf{x}_v}}{\text{maximize}} \quad \alpha \sum_{i=1}^{N} \sum_{j=1}^{N_{\text{VNF}}} V_{i,j} - (1-\alpha) \sum_{j=1}^{N_{\text{VNF}}} D_j \tag{P1}$$

$$\text{subject to} \quad \sum_{i=1}^{N} V_{i,j} \leq 1,\ j = 1, \ldots, N_{VNF}, \tag{P1.a}$$

$$x_{ol,i} + x_{of,i} + x_{v,i} \leq 1,\ \forall i = 1, \ldots, N, \tag{P1.b}$$

$$\sum_{i=1}^{N} x_{e,i} = 2p + 1,\ p \in \mathbb{N}, \tag{P1.c}$$

$$\sum_{i=1}^{N} x_{ol,i} = 1, \tag{P1.d}$$

$$\sum_{i=1}^{N} x_{of,i} = 2p' + 1,\ p' \in \mathbb{N}, \tag{P1.e}$$

$$\sum_{i=1}^{N} x_{v,i} = 2p'' + 1,\ p'' \in \mathbb{N}, \tag{P1.f}$$

$$x_{v,i} * G * C_b \leq R_i^{\text{stor}},\ \forall i = 1, \ldots, N, \tag{P1.g}$$

$$\sum_{j=1}^{N_{\text{VNF}}} V_{i,j} I_{\text{UE,j}}^{\text{CPU}} \leq R_i^{\text{CPU}} (1 - \sum_{\substack{k=1,\ldots,4 \\ \to k' \in \{e,ol,of,v\}}} \beta_k x_{k',i}), \tag{P1.h}$$

$$x_{v,i} - x_{e,i} \geq 0,\ \forall i = 1, \ldots, N \tag{P1.i}$$

where $\alpha \in [0,1]$ in the objective function is a weighting factor that balances between maximizing the successful placement of

VNF instances and minimizing the blockchain delay (i.e., the time needed to add a new block relative to the user VNF requests as transactions). Constraint (P1.a) ensures that if a user VNF j is approved for deployment, it will be allocated to a single gNB, while (P1.b) guarantees the decentralization of the blockchain functions, where each gNB can host at most only one type of blockchain VNF. Also, (P1.c)–(P1.f) impose that the numbers of endorsers, orderer-followers, and validators are respectively odd, and there is a single orderer-leader. This requirement facilitates decision-making processes by minimizing the risk of tie outcomes, which could lead to indecision or stalemates. Constraint (P1.g) ensures that if a gNB i is selected to host a validator VNF, it must possess adequate storage capacity to accommodate the entire ledger and its future expansion. Note that G in (P1.g) represents the maximal number of blocks growth in the ledger, while C_b denotes the size of a single block. Also, constraint (P1.h) guarantees that the cumulative computational needs of all VNFs, including those for blockchain and users on the gNB i, do not exceed the node's peak CPU capability. Finally, given that $\mathcal{S}_e \subset \mathcal{S}_v$, it is required that all nodes in $\mathcal{S}_v$ possess a VNF for validation, while nodes in $\mathcal{S}_e$ deploy both the endorser and validator VNFs, as shown in (P1.i).

Problem (P1) can be easily assimilated to an instance of VNF placement problem (regardless of the blockchain or user nature of the VNFs), which is known to be NP-hard [13]. Hence, by restriction, our problem is also NP-hard and cannot be solved in a polynomial time.

V. Proposed VNF Placement Algorithms

Given the NP-hardness of problem (P1), we propose to decompose it into two sub-problems. First, we address in (P2) the placement of blockchain VNFs with the goal of minimizing the total block validation delay. Then, we focus on the user VNF placement problem to maximize successful user VNFs deployment in (P3). Subsequently, the first subproblem (P2) can be formulated as:

$$\underset{\substack{\mathbf{x}_e,\mathbf{x}_{ol},\\ \mathbf{x}_{of},\mathbf{x}_v}}{\text{minimize}} \quad \sum_{j=1}^{N_{\text{VNF}}} D_j \tag{P2}$$
$$\text{subject to} \quad (P1.b)-(P1.g),(P1.i),$$

while the second subproblem (P3) is written as:

$$\underset{\mathbf{V}}{\text{maximize}} \quad \sum_{i=1}^{N}\sum_{j=1}^{N_{\text{VNF}}} V_{i,j} \tag{P3}$$
$$\text{subject to} \quad (P1.a),(P1.h).$$

To solve these consecutive sub-problems with low time complexity, we propose the following solutions:

A. PSO-based VNF placement

PSO is a meta-heuristic optimization method for converging a set of local solutions (*pBest*) to a global solution (*gBest*) [14]. Each solution is represented by a particle characterized by its position and velocity. Over the course of iterations, the particles move by modifying their position and speed until they converge to a global solution. We apply PSO to our problem as follows. We consider the swarm of particles to be the set of $\mathcal{N}$ physical nodes in the network, i.e., gNBs. In our approach, we aim to identify the best nodes to act as blockchain nodes, i.e., those that minimize communication delays via the UAV. Each PSO particle represents a subset of blockchain nodes $\mathcal{S}$ among the set $\mathcal{N}$. At each iteration and for each particle, we determine the best combination of blockchain nodes that minimizes the resulting sum delay. This is achieved by calling the objective function while adhering to the constraints of (P2). Next, the PSO algorithm compares the delays of the best combinations from the different particles (*pBest*) to determine the global best solution (*gBest*) for the particles' swarm. Finally, the algorithm calculates the particles' velocities using (14)–(15) and updates their positions with (16)–(17) below.

$$V_i^{\theta+1}(x) = \omega.V_i^{\theta}(x) + c_1.r_1.(gBest(x) - X_i^{\theta}) + c_2.r_2.(pBest(x) - X_i^{\theta}) \tag{14}$$

$$V_i^{\theta+1}(y) = \omega.V_i^{\theta}(y) + c_1.r_1.(gBest(y) - Y_i^{\theta}) + c_2.r_2.(pBest(y) - Y_i^{\theta}) \tag{15}$$

$$X_i^{\theta+1}(x) = X_i^{\theta}(x) + V_i^{\theta+1}(x) \tag{16}$$

and

$$X_i^{\theta+1}(y) = X_i^{\theta}(y) + V_i^{\theta+1}(y) \tag{17}$$

Note that ($\mathbf{X}_i^{\theta}$, $\mathbf{Y}_i^{\theta}$) and $\mathbf{V}_i^{\theta}$(x,y) represent the coordinates of the particle's center and its velocity at iteration θ_i, respectively. c_1 and c_2 are acceleration coefficients that weight the influence of *pBest* and *gBest*, r_1 and r_2 are uniformly distributed random variables between 0 and 1, and ω is the inertia weight that controls the influence of the previous velocity. According to [15], it is preferable to initialize ω with a high value and decrease it gradually to obtain refined solutions. It is then expressed by

$$\omega = \omega_{\text{max}}.\frac{(\omega_{\text{min}} - \omega_{\text{max}})}{\theta_{\text{max}}} \cdot \theta_i, \tag{18}$$

where ω_{max} and ω_{min} are the maximum and minimum inertia weight values, fixed in our simulations to 0.9 and 0.4, respectively. To ensure the algorithm runs within a reasonable time, we set the maximum number of iterations to θ_{max} = 100.

The same methodology is used to solve problem (P3). Specifically, PSO determines the order of VNFs' deployment on the gNBs to ensure maximal deployment. Each particle is represented by a deployment order (*p'Best*) and its velocity. Through iterations, the particles adjust their deployment order to converge towards an optimal order *g'Best*. The initial number of particles is set to 25 and the maximum number of iterations θ'_{max}= 100. Finally, the pseudo-code of our PSO-based solution is presented in Algorithm 1.

B. Greedy-based VNF placement

In this approach, we proceed as follows. We start by solving (P2). Specifically, we sort the blockchain nodes in descending order based on their total computing consumption and the data size transferred during a single process. Next, we sort the physical nodes in descending order based on their scores

$$Score_i = 0.6 * \frac{R_i^{\text{CPU}} - R_{\text{min}}^{\text{CPU}}}{R_{\text{max}}^{\text{CPU}} - R_{\text{min}}^{\text{CPU}}} - 0.4 * \frac{R_i^{\text{GAIN}} - R_{\text{min}}^{\text{GAIN}}}{R_{\text{max}}^{\text{GAIN}} - R_{\text{min}}^{\text{GAIN}}}, \ \forall i \in \mathcal{N}, \tag{19}$$

Algorithm 1: Proposed PSO-based Algorithm

```
    Input: $\mathcal{N}$ and PSO parameters
    Output: $gBest,\ gDealy,\ g'Best,\ g'Max_{\text{vnf}}$
1   Initialize list of particles $\mathcal{P}$ and velocity $v(x,y)$ ;
2   $gBest \leftarrow \emptyset$ ;
3   $gDelay \leftarrow +\infty$ ;
4   while $\theta_i < \theta_{max}$ do
5       for each $p \in \mathcal{P}$ do
6           $pBest,\ pDelay \leftarrow cost\_function(p)$ from (P2);
7           if $pDelay \leq gDelay$ then
8               $gBest \leftarrow pBest$ ;
9               $gDelay \leftarrow pDelay$ ;
10      Update inertia weight $\omega$ ;
11      for each $p \in \mathcal{P}$ do
12          Update velocity $v(x,y)$ of $p$ ;
13          for each $node \in p$ do
14              Update position of $node$ ;
15              if the new node coordinates are outside the swarm then
16                  Correct new coordinates ;
17      $\theta_i = \theta_i + 1$
18  Subtract the CPU allocated to the blockchain from the physical nodes;
19  initialize list of particles $\mathcal{P}'$ and velocity $v'(x,y)$ ;
20  $\mathcal{N}'_{VNF} \leftarrow \mathcal{N}_{VNF}$ sorted in ascending order of CPU consumption;
21  $g'Best \leftarrow \emptyset$ ;
22  $g'Max_{\text{vnf}} \leftarrow 0$ ;
23  while $\theta'_i < \theta'_{max}$ do
24      for each $p' \in \mathcal{P}'$ do
25          $pMax_{\text{vnf}}, g'Best \leftarrow cost\_function'(p')$ using (P3);
26          if $p'Max_{\text{vnf}} >\ g'Max_{\text{vnf}}$ then
27              $g'Best \leftarrow p'Best$ ;
28              $g'Max_{\text{vnf}} \leftarrow p'Max_{\text{vnf}}$ ;
29      Update inertia weight $\omega$' ;
30      for each $p' \in \mathcal{P}'$ do
31          Update velocity $v'(x,y)$ of $p$ ;
32          for each $node \in p'$ do
33              update position of $node$ ;
34      $\theta'_i = \theta'_i + 1$ ;
35  return $gBest,\ gDelay,\ g'Best,\ g'Max_{\text{vnf}}$;
```

where R_i^{GAIN} is the channel gain of the gNB i. $R_{\max}^{\text{CPU}}, R_{\min}^{\text{CPU}}, R_{\max}^{\text{GAIN}}, R_{\min}^{\text{GAIN}}$ are the maximum and the minimum CPU capacity and channel gain that a gNB can have in our system, respectively. Then, we assign the first blockchain node from $\mathcal{S}^{'}$ to the first node in $\mathcal{N}^{'}$, ensuring that two blockchain nodes are not deployed on a single physical node.

Algorithm 2: Greedy-based Algorithm

```
    Input: $\mathcal{S}$, $\mathcal{N}$, $\mathcal{N}_{VNF}$
    Output: $\mathbf{V}, \mathbf{x}_e, \mathbf{x}_{ol}, \mathbf{x}_{of}, \mathbf{x}_v$
1   $\mathcal{S}^{'} \leftarrow \mathcal{S}$ sorted in descending order of total CPU power consumption and total transferred data size;
2   $\mathcal{N}^{'} \leftarrow \mathcal{N}$ sorted in descending order based on eq.(19);
3   $i \leftarrow 1$ ;
4   for each $s \in \mathcal{S}'$ do
5       $x_{k',\mathcal{N}'(i)} \leftarrow 1$, with $k' \in \{e, ol, of, v\}$ depending on $s$;
6       $i \leftarrow i + 1$ ;
7   Subtract the CPU allocated to the blockchain from the physical nodes;
8   $\mathcal{N}'' \leftarrow \mathcal{N}$ sorted with the new CPU capacities;
9   $\mathcal{N}_{VNF}^{'} \leftarrow \mathcal{N}_{VNF}$ sorted in ascending order of CPU consumption;
10  $j \leftarrow 1$;
11  for $n \in \mathcal{N}''$ do
12      while CPU Requirement of $\mathcal{N}_{VNF}^{'}(j) \leq$ Remaining capacity of $n$ AND $j \leq len(\mathcal{N}_{VNF}^{'})$ do
13          $V_{n,j} \leftarrow 1$;
14          $j \leftarrow j + 1$ ;
15  return $\mathbf{V}, \mathbf{x}_e, \mathbf{x}_{ol}, \mathbf{x}_{of}, \mathbf{x}_v$ ;
```

To solve (P3), we deduce the computing power assigned to the blockchain. Next, we sort the physical nodes based on their remaining computing capacity in descending order and sort the user VNF requests based on their CPU requirements in ascending order. For each ordered gNB, we allocate user VNFs until there is no more available space. This process continues until all gNBs are saturated. Finally, the greedy-based solution is summarized in Algorithm 2.

VI. Performance Evaluation

In this section, we gauge the effectiveness of our proposed resource allocation algorithms based on extensive simulations. First, we describe the simulation environment setup. Then, we analyze the obtained results and discuss the effectiveness of our proposals compared to: i) the optimal approach using the GUROBI solver [16], and ii) the random approach.

A. Simulation setup

We consider a dense urban area of 2×2 km^2, where communications experience attenuation with $\zeta = 0.2$ and environment parameters $a = 15$ and $b = 0.5$ [17]. The reference channel gain is $g_0 = -50$ dB, the communication bandwidth is $B = 20$ MHz [17], and the unitary noise power $\sigma_{u,i} = -174$ dBm/Hz. In our system, $N = 154$ gNBs are deployed, where each of them has a coverage of 100 m. The CPU capacity of a gNB is uniformly distributed and represented by $R_i^{\text{CPU}} \in [4,6]$ GHz, and the storage capacity is $R_i^{\text{stor}} \in [100,200]$ Gigabytes (GB) [18]. The gNBs have heights $H_i = 10$ m and transmit powers $P_i = 33$ dBm, $\forall i = 1,\ldots,N$. The UAV's coordinates

TABLE II
BLOCKCHAIN PARAMETERS' VALUES

Parameter	Value
I_e^{CPU}	500000000 cycle/transaction
I_{ol}^{CPU}	728000000 cycle/block [19]
I_{of}^{CPU}	182000000 cycle/block [19]
I_v^{CPU}	300000000 cycle/transaction
$\beta_1, \beta_2, \beta_3, \beta_4$	0.3, 0.3, 0.3, 0.3
$W_{u,e}$	500 kilobytes (KB) [20]
$W_{e,u}, W_{u,ol}$	2000 KB [20]
$W_{ol,u}, W_{u,of}, W_{u,v}$	2800 KB [20]
$W_{of,u}, W'_{u,ol}, W_{v,u}$	1500 Bytes [20]

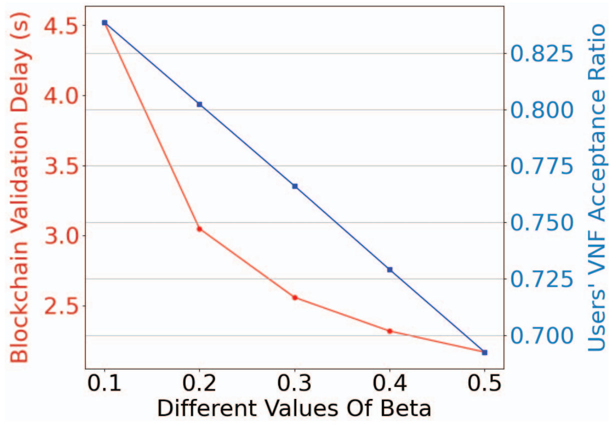

Fig. 3. Blockchain validation delay and users' VNF acceptance ratio vs. β (Optimal).

are $\mathbf{q}_u = [1000, 1000, 150]$. It hovers at the center of the area and transmits at power $P_u = 23$ dBm. To effectively monitor the system, we deploy 47 blockchain nodes, such that $n_e = 15$, $n_{ol} = 1$, $n_{of} = 15$, and $n_v = 31$, aiming to handle $N_{\text{VNF}} = 6000$ users' VNF requests, where the computing resource consumption of a VNF of user j is $I_{\text{UE,j}}^{\text{CPU}} \in [100, 200]$ megahertz (MHz). Moreover, we set the required ledger size to 130 GB. The blockchain transactions are simulated based on the flow of Hyperledger Fabric network and RAFT consensus, as shown in Fig. 2. Also, the data size transferred during the blockchain validation process and the computing power consumed in each phase are derived from the official Hyperledger Fabric documentation and previous research works, as shown in Table II.

B. Simulation results

Before presenting the performances of the proposed approaches, we show in Fig. 3 the reference results in terms of blockchain validation delay and users' VNF acceptance ratio of the optimal solution, provided using the GUROBI solver, and for different $\beta = \beta_1 = \ldots = \beta_4$ values. β represents the needed gNB computing power to support blockchain VNFs. As β increases, the users' VNF acceptance ratio degrades from 84% to 69%, and the blockchain delay improves from 4.52 sec to 2.17 sec. This is expected since more CPU power is allocated to the blockchain services, thus reducing the blockchain operations delay. However, this comes at the expense of reducing the users' VNFs acceptance ratio. Also, for $\beta = 0.3$ (the median value), we observe that the users' VNF acceptance ratio is at a moderate level, and at the same time, we are 16% closer to the minimum blockchain validation delay. Hence, we set $\beta = 0.3$ for all subsequent simulations.

TABLE III
BLOCKCHAIN VALIDATION DELAY VS. NUMBER OF TRANSACTIONS PER BLOCK (DIFFERENT METHODS)

Nb. Trans. per Block	Optimal	PSO	Greedy	Random
1	2.56 s	2.62 s	2.62 s	2.76 s
10	20.94 s	21.33 s	21.36 s	22.37 s
100	204.3 s	208.04 s	208.78 s	219.96 s
1000	2038.77 s	2076.13 s	2083.82 s	2167.84 s

TABLE IV
USERS' VNF ACCEPTANCE RATIO VS. NUMBER OF TRANSACTIONS PER BLOCK (DIFFERENT METHODS)

Nb. Trans. per Block	Optimal	PSO	Greedy	Random
Any	81.7 %	81 %	80.5 %	76.8 %

Table III presents the blockchain validation delay for different numbers of transactions per block. We can see that as the number of transactions per block increases, the delay grows exponentially for all methods due to the longer ordering processing time needed. The random approach achieves the longest blockchain validation times, while the proposed greedy and PSO-based methods realize close delays to those of the optimal method.

In Table IV, we present the users' VNF acceptance ratio results related to Table III. We observe that the acceptance ratio remains stable regardless of the number of transactions per block for all methods. This is because the CPU consumption rate of the physical nodes assigned to the blockchain VNFs (i.e., β) remains constant in our experiments. Moreover, both the PSO and greedy methods provide near-optimal results, with slight degradation of only 0.86% and 1.47% compared to the optimal one. in contrast, they are significantly faster than the optimal approach, as illustrated in Table V.

Fig. 4 depicts the blockchain throughput (i.e., the number of transactions validated per second) as a function of the number of transactions per block for all methods. We see that the throughput increases initially, but then remains stable. Indeed, as we increase the number of transactions per block from 1 to 10, to 100, and 1000, the blockchain validation delay rises by 8.1, 9.7, and 10 times, respectively. Also, both greedy and PSO methods achieve very similar throughputs, which are slightly lower than the optimal one, but better than the random benchmark.

Finally, Fig. 5 presents the impact of the number of blockchain nodes on the blockchain validation delay and the users' VNF acceptance ratio. We observe that the blockchain delay increases with the number of nodes in the blockchain slice. In contrast, the users' VNF acceptance ratio decreases. This is simply because the new blockchain nodes may be deployed on gNBs with less CPU power and/or located farther

TABLE V
EXECUTION TIMES TO SOLVE PROBLEMS (P2) AND (P3) FOR DIFFERENT METHODS

Execution time	Optimal	PSO	Greedy	Random
(P2)	90 min	15.9 s	0.006 s	3.8 s
(P3)	360 min	38.8 s	0.004 s	1.27 s

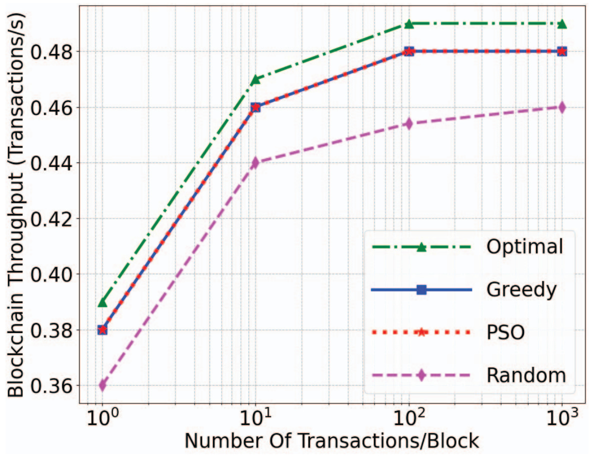

Fig. 4. Blockchain Throughput vs. Nbr. of Transactions per block.

from the UAV compared to previous nodes. In addition, the increasing number of blockchain nodes consumes more physical resources, thus leaving fewer resources for users' VNFs. Although both PSO and greedy approaches realize very similar blockchain validation delays, the PSO achieves a 10% improvement in terms of VNF acceptance ratio compared to the greedy method.

VII. Conclusion

In this paper, we addressed the potential application of blockchain to improve the privacy and security of Internet of Vehicle (IoV) systems. This involves verifying the integrity and authenticity of service requests from IoV nodes through an access control policy implemented as a smart contract. Specifically, we formulated the VNF resource placement optimization of both blockchain and users' services as an ILP problem, whose objective is to maximize the acceptance rate of users' service requests while guaranteeing a minimal delay for blockchain operations to validate those requests. We then proposed two heuristic algorithms to solve this NP-hard problem, namely the PSO and greedy approaches. Simulation results showed that both PSO and greedy solutions achieve similar and near-optimal blockchain validation delays. However, the greedy heuristic realizes a 10% lower VNF acceptance ratio compared to PSO, but with a negligible computation time.

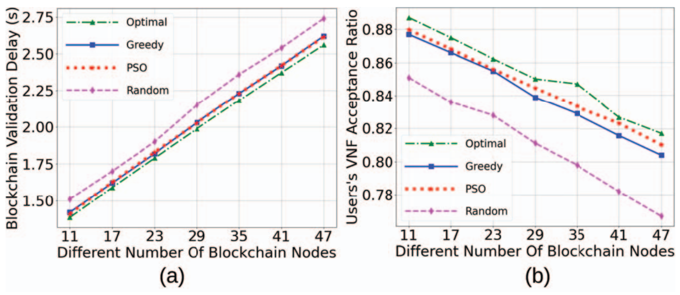

Fig. 5. Impact of nbr. of blockchain nodes. (a) Blockchain validation delay; (b) Users' VNF acceptance ratio.

References

[1] Y. Xu, E. Yu, Y. Song, F. Tong, Q. Xiang, and L. He, "$\mathcal{R}$-tracing: Consortium blockchain-based vehicle reputation management for resistance to malicious attacks and selfish behaviors," *IEEE Trans. Veh. Technol.*, vol. 72, no. 6, pp. 7095–7110, 2023.

[2] Y. Liu, M. Xiao, S. Chen, F. Bai, J. Pan, and D. Zhang, "An intelligent edge-chain-enabled access control mechanism for iov," *IEEE Internet of Things Journal*, vol. 8, no. 15, pp. 12 231–12 241, 2021.

[3] X. Wang, H. Zhu, Z. Ning, L. Guo, and Y. Zhang, "Blockchain intelligence for internet of vehicles: Challenges and solutions," *IEEE Commun. Surv. Tuts.*, vol. 25, no. 4, pp. 2325–2355, 2023.

[4] M. Shen, H. Lu, F. Wang, H. Liu, and L. Zhu, "Secure and efficient blockchain-assisted authentication for edge-integrated internet-of-vehicles," *IEEE Trans. on Vehicular Technology*, vol. 71, no. 11, pp. 12 250–12 263, 2022.

[5] A. Samy, I. A. Elgendy, H. Yu, W. Zhang, and H. Zhang, "Secure task offloading in blockchain-enabled mobile edge computing with deep reinforcement learning," *IEEE Trans. on Network and Service Management*, vol. 19, no. 4, pp. 4872–4887, 2022.

[6] S. Roy, S. Nandi, R. Maheshwari, S. Shetty, A. K. Das, and P. Lorenz, "Blockchain-based efficient access control with handover policy in iov-enabled intelligent transportation system," *IEEE Trans. on Vehicular Technology*, vol. 73, no. 3, pp. 3009–3024, 2024.

[7] X. Ye, M. Li, P. Si, R. Yang, Z. Wang, and Y. Zhang, "Collaborative and intelligent resource optimization for computing and caching in iov with blockchain and mec using a3c approach," *IEEE Trans. on Vehicular Technology*, vol. 72, no. 2, 2023.

[8] N. Khan, A. Ahmad, A. Wakeel, Z. Kaleem, B. Rashid, and W. Khalid, "Efficient uavs deployment and resource allocation in uav-relay assisted public safety networks for video transmission," *IEEE Access*, vol. 12, pp. 4561–4574, 2024.

[9] D. Ongaro and J. Ousterhout, "In search of an understandable consensus algorithm," in *2014 USENIX annual technical conference (USENIX ATC 14)*, 2014, pp. 305–319.

[10] Hyperledger Fabric, "Introduction," 2023, accessed: 2024-06-25. [Online]. Available: https://hyperledger-fabric.readthedocs.io/en/release-2.5/whatis.html

[11] S. Nakamoto. Bitcoin:, "A peer-to-peer electronic cash system."

[12] V. B. et al., "A next-generation smart contract and decentralized application platform," *white paper*, p. 3(37), 2014.

[13] F. Bari, S. R. Chowdhury, R. Ahmed, R. Boutaba, and O. C. M. B. Duarte, "Orchestrating virtualized network functions," *IEEE Transactions on Network and Service Management*, vol. 13, no. 4, pp. 725–739, 2016.

[14] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, 1995, pp. 39–43.

[15] A. Mseddi, W. Jaafar, H. Elbiaze, and W. Ajib, "Joint container placement and task provisioning in dynamic fog computing," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 10 028–10 040, 2019.

[16] G. Optimization, "State-of-the-art mathematical programming solver," Online, Apr 2014, available: http://www.gurobi.com/.

[17] X. Dai, Z. Xiao, H. Jiang, and J. C. S. Lui, "Uav-assisted task offloading in vehicular edge computing networks," *IEEE Trans. on Mobile Computing*, vol. 23, no. 4, pp. 2520–2534, 2024.

[18] T. Liu, S. Ni, X. Li, Y. Zhu, L. Kong, and Y. Yang, "Deep reinforcement learning based approach for online service placement and computation resource allocation in edge computing," *IEEE Trans. on Mobile Computing*, vol. 22, no. 7, pp. 3870–3881, 2023.

[19] G. Yang, K. Lee, K. Lee, Y. Yoo, H. Lee, and C. Yoo, "Resource analysis of blockchain consensus algorithms in hyperledger fabric," *IEEE Access*, vol. 10, pp. 74 902–74 920, 2022.

[20] Hyperledger Fabric, "Performance and scalability," 2023. [Online]. Available: https://hyperledger-fabric.readthedocs.io/en/release-2.5/performance.html