






# Traffic Flow Optimization for UAVs in Multi-Layer Information-Centric Software-Defined FANET

Liehuang Zhu , Senior Member, IEEE, Md Monjurul Karim , Kashif Sharif , Senior Member, IEEE, Chang Xu , Member, IEEE, and Fan Li , Member, IEEE

**Abstract**—Unmanned Aerial Vehicles (UAVs) have received significant research interest from academia due to their on-demand content distribution capabilities using mobile edge computation and the next-generation Flying Ad-hoc Network (FANET). With the addition of Software-Defined Networking (SDN) and network virtualization, these UAVs have transformed into three-dimensional distributed heterogeneous networks. However, the softwareized UAV-based communication is prone to high latency, energy consumption, resource constraints, and link failures. Hence, content orchestration has become a significant challenge. Information-Centric Networking (ICN) uses content-based rapid data dissemination in the dynamic wireless scenario. However, ICN-based content discovery and distribution have not been explored extensively for UAV-assisted networks. In this work, we propose a UAV-assisted multi-layer IC-SDN solution to tackle the content distribution challenges using distributed controllers placed hierarchically in the edge and cloud tiers. Besides, we formulate the traffic optimization problem into a joint forwarding and flow scheduling problem using M/M/1 queueing allocation model and propose a heuristic edge-cloud traffic flow assignment solution that allocates requests based on the service type and device location. We evaluate the proposed solution in a simulation environment considering the mobility principle of FANET nodes. Besides, the effectiveness of the optimization solution and the performance gains are evaluated analytically. The simulation and numerical results show that the proposed optimization model is efficient as compared to other solutions, in maximizing throughput and minimizing computational latency, delay, and packet loss.

**Index Terms**—Flow optimization, information-centric networking, software-defined networks, UAVs.

## I. INTRODUCTION

THE modern internet is driven by the synthesis of Fifth-Generation (5G) wireless network, Artificial Intelligence

Manuscript received 4 December 2021; revised 13 August 2022; accepted 18 September 2022. Date of publication 10 October 2022; date of current version 13 February 2023. This work was supported in part by the National Natural Science Foundation of China under Grants 61972037, 61872041, and U1804263 and in part by Shandong Provincial Key Research and Development Program under Grants 2021CXGC010106. The review of this article was coordinated by Prof. Yi Qian. (Corresponding author: Kashif Sharif.)

Liehuang Zhu and Chang Xu are with the School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing 100081, China (e-mail: liehuangz@bit.edu.cn; xuchang@bit.edu.cn).

Md Monjurul Karim is with the School of Computer Science and Technology, Beijing Institute of Technology, Beijing 100081, China (e-mail: mkarim@bit.edu.cn).

Kashif Sharif and Fan Li are with the School of Computer Science and Technology, Beijing Institute of Technology, Beijing 100081, China, and also with the Beijing Engineering Research Center of High Volume Language Information Processing and Cloud Computing Applications, Beijing Institute of Technology, Beijing 100081, China (e-mail: kashif@bit.edu.cn; fli@bit.edu.cn).

Digital Object Identifier 10.1109/TVT.2022.3213040

(AI), Internet of Things (IoT), and edge computing [1]. These technologies potentially increase the capabilities of participating devices in wireless ad-hoc environment to initiate effective communication with neighboring nodes while satisfying the end-users by transferring massive data in a few milliseconds [2]. In recent years, academia and industry have strived to innovate efficient wireless and infrastructure-less ad-hoc communication that utilizes edge computing, and 5G networks [3]. Unmanned aerial vehicles (UAVs), also known as drones, have emerged as an exciting new area that forms a unique type of ad-hoc communication titled flying ad-hoc networks (FANETs) [4]. In general, UAVs are classified into low-altitude platforms (LAP) and high-altitude platforms (HAP) based on their application, weight, range, flying principle, resource utilization, and energy consumption [5]. Besides, a subset of UAVs form a swarm and communicate with each other. Some of them communicate directly to the infrastructure on the ground, while others communicate with the intermediate or neighboring UAVs through multi-hop communication.

Compared to a mobile ad-hoc networks (MANET) and vehicular ad-hoc networks (VANET), the mobility of UAVs in a FANET is significantly different in terms of speed and degree (i.e., 3D) in the air. For example, in an inter-UAV network, UAVs generally operate at speeds of 20 to 500 km/hour to share information, participate in the rescue mission, and maintain awareness of targeted objects in the air and ground [6]. Therefore, UAVs in FANET are required to maintain several key characteristics such as diversified QoS requirements, dynamic topology changing, higher mobility, inadequate resources, low frequency node, and consistent network partitioning that introduce link disruption, node failure, routing, and computation overhead in the network [7], [8], [9], [10], [11], [12].

SDN and NFV emerge as promising solutions [13], [14] to tackle some of the FANET-based challenges [15], [16], [17]. The control plane in an SDN-based UAV can either be implemented separately or merged with the data plane. The majority of contributions regarding softwareized UAV-based architecture focus on mobility, routing, monitoring, controller placement, security enforcement, and coordination with VANET, IoT, and Wireless Sensor Networks (WSN) [18], [19], [20]. Hence, the mobility of UAV nodes introduces several challenges in SDN-based scenarios [21]. Compared to mobile wireless ad hoc nodes, UAV rotates the current state's position to the changed state at a minimum of 100 to 500 m/s; 5 to 10 times faster than VANET and 100 times faster than MANET. Therefore,

lack of efficient and scalable communication between SDN controllers and UAV may lead to participating components, i.e., sensing unit, base station, radio-access tower, backbone network, and back-end server (e.g., data center) susceptible to congestion, delay, traffic consumption, and bandwidth-related bottlenecks [5].

ICN is a promising communication paradigm that replaces the traditional host-dependent end-to-end communication into name-based hop-by-hop communication emphasizing on content first location second principle [22], [23]. ICN performs in-network caching, named-based forwarding, and mobility management, allowing infrastructure-less wireless ad-hoc communication (e.g., IoT, WSN, FANET) to achieve low latency and minor network load. In addition, ICN offers several advantages in the UAV-assisted FANET [24], [25], [26], [27], [28]. First, a flexible namespace offers unique identification of contents generated in different heterogeneous layers of the networks [26]. For example, contents within a geographical area of interest can be aggregated or retrieved through data-centric model and enforced queries with a single interest or request with a hierarchical namespace. Such namespace structure includes geographical references (e.g., GPS details), service types, devices, or sensor IDs [25]. Besides, ICN also supports asynchronous information dissemination through publish-subscribe (pub-sub) mechanisms (e.g., pull and push-based) which allows UAVs to manage their subscription requests and other records while shifting position from one access point to another in the network. For example, an enhanced pull-based pub-sub approach is presented in [25] that ensures critical information retrieval by UAVs when there are temporary interruptions in the network. Furthermore, in-network caching enhances the previously requested contents to make them available at specific access points (e.g., base stations and ground control stations), resulting in limited bandwidth usage and lower latency. More specifically, specific access points acting as border gateways maintain their cached contents by keeping the freshness of local data and updating them consistently [24].

By acknowledging the benefits of both SDN and ICN, we aim to integrate them into an IC-SDN communication principle that offers both flow-based and named-based forwarding and caching into the FANET system while preserving the UAV-specific QoS requirements in the network. However, implementing a content-oriented networking flow into the same or different domains would require path optimization, multipath forwarding, and allocating specific services to different nodes. In FANET, the nodes are diversified as UAVs, ground units, road-side units (RSUs), base stations (BSs), radio access networks, and remote data centers have unique principles and characteristics. Edge computation can solve latency, throughput, system cost, delay, and signaling overhead by bringing the content closer to the consumer. Therefore, integrating edge computing capabilities on the controller would allow an automated discovery and distribution of contents regardless of the service type and the location of consumer and producer nodes. As a result, applying joint optimization techniques becomes a promising method to tackle the challenges of data dissemination, coverage, mobility, and energy efficiency while outlining a reliable communication

solution that effectively integrates communication, caching, and computing in FANET [29].

Therefore, in this paper, we adopt an IC-SDN principle that allows the controllers located in the edge and cloud to perform name-based flow installation in the aerial and ground units with in-network caching capability while acknowledging the challenges mentioned earlier. The controller keeps track of the available contents in both air and ground layers and guides the consumer node from the ground layer to retrieve the most desired content within the same or different domains using the most optimal path. A software-defined FANET system with a multi-layered control plane is designed and analytically modeled to perform these operations. In addition, the proposed communication and queue allocation model utilizes named-based information sharing among the participated nodes and allocates tasks based on the severity of service requested by the air and ground units.

The major contributions of this work are given below.

- A multi-layered information-centric softwareized framework is proposed to integrate UAVs, sensor nodes, and SDN controllers with edge and cloud services. Based on QoS requirements, the system initializes the automated content distribution and traffic allocation over edge and cloud layers.
- A hybrid combined pull and push-based information-centric communication paradigm has been proposed using the hierarchical placement of multiple controllers to exploit the simultaneous advantages of edge computation and network virtualization.
- UAVs are utilized to collect necessary information from the sensing units under a particular range and offload additional cost computation. Hence, UAVs act as mediators between the data units (sensing nodes) and the control units (edge and cloud controllers).
- The joint forwarding and flow scheduling problem over a multi-layer air-ground network is formulated analytically. As the optimization problem is NP-hard, the problem is decomposed into two-layered (i.e., edge and cloud) sequential flow assignments. In addition, a heuristic solution is applied to each of these assignments to provide rigorous bounds on the computational complexity and to allow the packet retrieving procedure in a UAV-assisted environment in the most efficient way.
- The system model is prototyped into a simulated environment to showcase the feasibility and efficiency of the proposed method.

The remainder of the paper is organized into five sections. **Section II** outlines a generic overview of the softwareized FANET and then discusses the existing softwareized UAV-based traffic orchestration solutions. **Section III** introduces the proposed multi-layered architecture highlighting the specification, characteristics, and working principle of individual nodes located in different layers. We then discuss the intra and inter-communication among participating nodes and formulate them analytically through the queue allocation model. We propose two assignment algorithms for both generic and computational traffic flows in **Section IV**. **Section V** presents the experimental evaluation

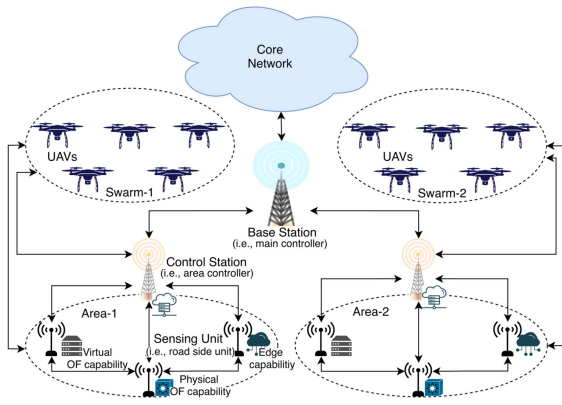


Fig. 1. Macro view of the topological model of softwareized UAV-assisted architecture.

and performance analysis of our solution comparing against state-of-the-art algorithms. Finally, we conclude the work in Section VI, recalling the primary contributions and discussing the future work.

## II. BACKGROUND AND RELATED WORK

We start by giving a short overview of softwareized FANET, including the main components and their working principle. Then, we discuss some of these existing solutions, and their significance and limitations in terms of architecture design and overall performance.

Fig. 1 shows the macro architecture of a softwareized UAV-based network. We use this specific model here, as it lays the ground topology for the proposed solution. UAVs operate individually or as swarm in a mesh-based topology. The main components of SDN-driven FANET are UAVs, sensing unit, control station, and base station. UAVs have multi-folded responsibilities in an SDN-driven FANET, where they act as clients, relay nodes, or on-board units (OBU) in the data plane. Besides, vehicles and other sensor devices act as middleware in some scenarios [12]. On the other hand, the sensing units in the ground behave as forwarding devices in the data plane and have OpenFlow (OF) [38] compatibility. The ground station works like a base station and initiates communication with UAVs in the sky from a certain distance [19], [30], [31].

In most cases, the SDN controller is initialized physically or virtually in the base station (BS), ground control station (GCS), smart vehicles, or embedded into the UAVs due to mobility and energy usage complexity [15], [39], [40], [41]. Regarding UAVs as SDN controllers, multiple UAVs create a group or swarm to establish a distributed hierarchical control plane where a single UAV performs as a master controller and others operate as domain controllers [42], [43]. In general, the SDN controller adopts a multi-path forwarding strategy using the diverse interface provided by UAVs and other sensor nodes. It then calculates the decoupled paths among those units based on their link status, quality, and available weight. When there is a link failure, the forwarding strategy offers alternative paths. Additionally, the 5G communication offers virtualization, slicing, and edge computing capabilities in the base station. Therefore, applying

edge capabilities to BS or control stations improves the overall performance and energy consumption, allowing the UAVs to fetch necessary information from a specific domain instead of connecting to the data center through cloud computing [44]. Authors in [32] propose an SDN-UAV framework where the UAVs are equipped with wireless interfaces allowing them to communicate with other nodes through OF protocol and southbound interfaces (SBI). Apart from these interfaces, the framework also features GPS, different sensors, and other computation and storage units. The solution is technically promising as it merges the SDN and NFV with a containerization technique that allows the lightweight inbuilt system to be integrated into UAVs on-demand resulting in lower energy consumption. However, the authors do not evaluate the proposed system against similar works. Secinti et al. [45] introduce fog computation into UAVs by optimizing the network and computational resources analytically in an SDN-based two-layered aerial mesh environment. The UAVs are implemented as intelligent computational nodes, whereas the controller has been given task allocation responsibilities on the mesh network based on the QoS requirements. Although the solution presents task allocation and queue processing among intra-UAVs, ground sensing units, and an SDN controller, it does not discuss the solution's energy consumption impact. On the other hand, implementing UAVs as fog nodes instead of an edge may introduce additional overhead in the network. In [19], UAVs are deployed as relay nodes (cloudlet) to help the MEC servers offloading computational and delay-sensitive tasks from the vehicles in a dynamic software-defined vehicular environment. Despite the theoretical analysis, cost-optimized algorithm, and performance evaluation favor the efficacy of the proposed protocol, the authors do not emphasize the SDN controller's impact on the solution. More specifically, they neglect some generic task implementation, such as collecting information from the topology and guiding the participating nodes to exchange information. The solution proposed in [46] adopts the ICN principle through blockchain. However, due to the consensus principle of blockchain, it introduces additional routing overhead and higher latency. As a result, the combination of SDN and ICN fits perfectly in the FANET infrastructure. The lack of a solution in this field motivates us to propose a solution that brings content closer to the UAV and maintains the scalability among the participating nodes, which is one of the fundamental requirements of a heterogeneous multi-layered network. Xiong et al. [36] present a combined hop-by-hop data aggregation with a store-carry forwarding module as a form of coalition formation to reduce additional energy in a multi-UAV system. Although the results show promising outcomes in energy consumption and several coalitions against hop limit and iteration time, the authors do not provide any details regarding the single point of failure and how the prototype can tackle node recovery and maintain scalable performance in the FANET ecosystem.

Table I compares our solution with existing softwareized solutions regarding objective, technology adopted, evaluation metrics and remarks on the key points. Most of the SDN-based UAV solutions neither address the controller-specific task assignment nor the coordination between distributed controllers. On the other hand, the information-centric UAV solutions do not

TABLE I  
COMPARISON OF EXISTING WORKS WITH PROPOSED SOLUTION

Ref.	Objectives	Techniques	Remarks
[19]	Task offloading	SDN, MEC	Depends on VANET.
[30]	Fast handover	SDN, 5G	UAVs depend on LOS; Higher deployment cost.
[31]	E2E data distribution	SDN	Centralized control; OF v.1.5 compatible; Multipath routing.
[32]	Monitoring	SDN, NFV	Configuration and deployment flexibility; Hypervisor and container application; ODL [33] and ONOS [34] compatibility.
[35]	Security enforcement	SDN, NFV	Higher memory consumption; Validated in real testbed.
[36]	Energy saving	SDN	Applicable to UAV swarms; Susceptible to single point failure.
[37]	Computation offloading; Route scheduling.	MEC, CC	Multi-hop routing; Latency is emphasized.
Proposed	Traffic optimization; Rapid data distribution	SDN, ICN, MEC, CC	Distributed control; Queue allocation; Traffic characterization.

MEC=Multi-Access Edge Computation, CC=Cloud Computation.

highlight the fundamental forwarding and caching capabilities of the sensor units or vehicles and how UAVs take advantage of in-network caching facilities from these nodes in the ground. Though these solutions tackle some of the existing issues, such as task assignment and routing orchestration theoretically, they do not focus on content distribution through traffic optimization. Therefore, we adopt a hybrid solution that takes advantage of both ICN and SDN, offering flow-based operation on the control plane while providing hop-by-hop forwarding into the data plane. Besides, the ICN-based packet structure is modified to take advantage of the pending interest table (PIT) that offers both on and off-path content caching depending on the service type and popularity of specific content. The synchronization between multiple controllers in different layers is optimized analytically, while edge computation facilitates improved data aggregation among UAVs and other units.

### III. INFORMATION-CENTRIC SOFTWAREZED FANET

In this section, we propose an information-centric softwarezied UAV-based framework that is divided into three identical layers as shown in Fig. 2. We discuss the solution from its architecture, system model and communication perspective below.

#### A. Architecture Overview

The proposed architecture is divided into ground, aerial, sub-control, and control layers. Similar to [18], both ground and aerial layer are placed under the data plane while sub-control and control layers are assigned under the control plane. The ground sensing units (GSUs) and UAVs are included in the ground and aerial layers, respectively. The sub-control layer

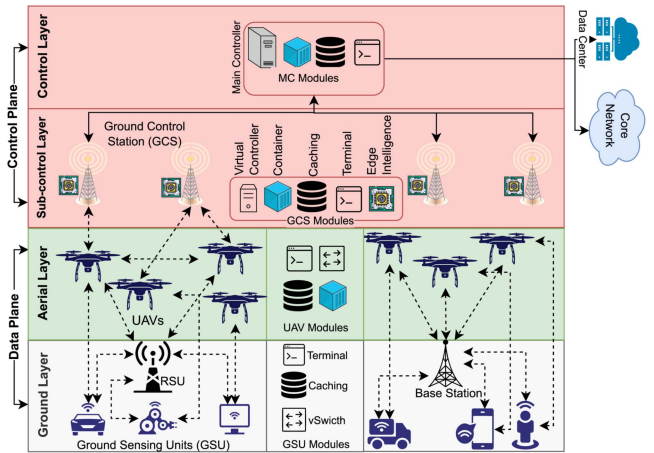


Fig. 2. Micro view of the proposed multi-layered UAV-assisted architecture.

includes the ground control station (GCS), while the main controller (MC) is placed in the control layer. Besides, the ground layer includes both stationary and non-stationary nodes. The stationary nodes (e.g., BSs, RSUs, or fixed sensing points) act as access or fixed data collecting points. Conversely, the non-stationary nodes (e.g., vehicles, IoT devices, cell devices, and other sensor devices) act as clients or consumers. In addition, we have different modules in each of these layers. For example, the virtualized switch (vSwitch) module allows the ground and aerial nodes to perform forwarding tasks. Similarly, the container module includes the operating system (OS) that provides an API to communicate with the GCS. Some of the OS-related functions in UAV would differ from GSU, GCS, and MC, as they have distinctive characteristics and service requirements in the proposed multi-layered architecture. Finally, we have assigned caching modules to all the participating nodes as we apply the ICN-based caching principle. Note that, UAVs are deployed as relay nodes that collect requests from stationary and non-stationary nodes in the ground layer and then forward them to GCS. In the existing ICN-enabled FANET or drone-based system, UAVs are utilized as forwarding nodes featuring three data structures, i.e., content store (CS), forwarding information base (FIB), and pending interest table (PIT) [24]. However, this is unsuitable for UAVs due to their limited energy and computation capability. Therefore, we remove FIB while modifying the PIT with a prefix of contents and interfaces (incoming and outgoing) to fit the context with the flow-based IC-SDN principle to satisfy the requirements for both FIB and PIT of the traditional ICN paradigm [47]. In addition, CS is also extended to improve the caching capability by introducing chunk-based content allocation that allows delivering more content with a single request.

The control layer in the middle is divided into two sublayers, i.e., control and the sub-control layer. a) The MC resides in the control layer, and it bridges the proposed architecture with the core network and remote data center that hosts various contents. b) Edge controllers are initialized in the sub-control layer as virtualized modules within the GCS placed into multiple regions. They perform virtualized network functions (i.e., traffic optimization, security enforcement), instructed by the MC. GCS

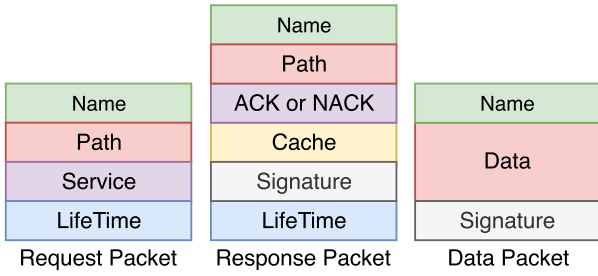


Fig. 3. Packet structures are defined for communication of system model.

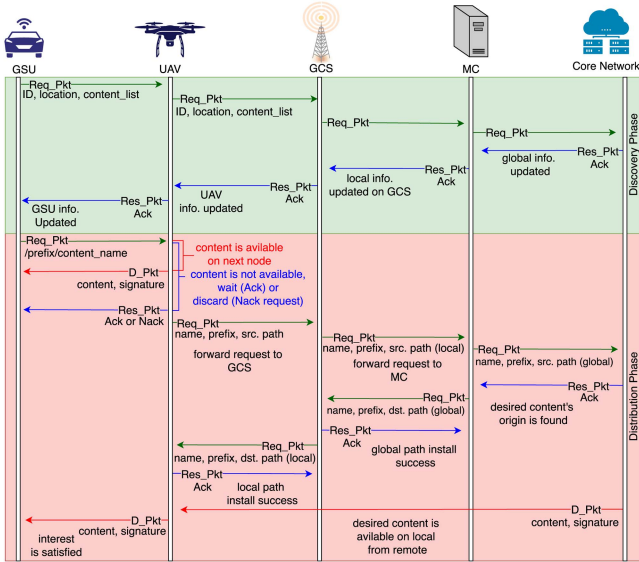


Fig. 4. Communication flow during content discovery and distribution phases.

communicates with the UAVs in the aerial layer to acquire necessary information regarding GSU and other UAVs. Afterward, it returns packets with flow instructions based on specific tasks and services to MC, which is later forwarded to UAVs on the aerial layer. Therefore, GCS acts as the most critical component in our system as it orchestrates necessary services between UAV, GSU, and MC.

**B. Communication Overview**

This section discusses the communication flow of our proposed framework that involves packet transmission between aerial, ground, and control layers, as shown in Fig. 4. Before giving the details, we first discuss the proposed packet structures used in the communication model. The communication model involving multiple hierarchically placed SDN controllers is motivated from our previous work in [48]. We divide the entire model into the discovery and distribution phase. These phases utilize three distinctive packet structures: Request Packet (*Req\_Pkt*), Response Packet (*Res\_Pkt*), and Data Packet (*D\_Pkt*), as shown in Fig. 3. The contents of the packets are derived from the Named Data Networking (NDN) [49] specification which have several commonalities with other ICN solutions.

*Req\_Pkt* carries four identical Type-Length-Value (TLV) headers: name, path, service, and lifetime. The name refers

to an ID of desired content that the UAV or GSU asks from GCS. Correspondingly, the path, service, and lifetime indicate a specific prefix, service types, and the duration of the *Req\_Pkt* on the receiving node, respectively. On the other hand, the *Res\_Pkt* contains name, path, acknowledgment, cache, signature, and lifetime. ACK has two possibilities: positive (ACK) or negative (NACK). When the *Res\_Pkt* carries ACK, the recipient node acknowledges that the requested service or content is under process. Conversely, NACK tells the recipient node to discard the request as the requested service cannot be processed further. Cache intrigues the receiving node to cache the data that it would receive from the remote location. Finally, the signature validates the signature of the data. Simultaneously, the receiving node verifies the content inside the data packet through the signature hints, and lifetime defines the duration the *Res\_Pkt* stays alive.

1) *Discovery Phase*: The discovery phase is critical to our proposed system as it allows the participating nodes to share necessary information once they join the topology. The information differs based on the characteristics of the participating nodes. For example, both UAVs and GSUs share generic information such as ID, prefix, bandwidth, cache, and energy capacity. On the other hand, velocity, payload capacity, trajectory, and altitude information are UAV-specific information. This phase also allows GCS and MC to learn about the local and global topology, respectively. Information is sent to GCS using a *Req\_Pkt*, while ACK is sent to requested nodes as a *Res\_Pkt* on successful initialization. If the transmitted packet fails to initiate the respective operation, GCS sends NACK to the requested node. As discussed previously, GSUs act as consumers while the UAVs act as forwarding nodes with or without caching capability. Meanwhile, the producer can be a remote content originator i.e., a data center located in the cloud or a core network that shares the original contents' source and signature information with MC. The location plays a vital role in our proposed system as it determines whether GSUs forward *Req\_Pkt* to UAVs or other fixed data collecting nodes on the ground, i.e., BS or RSU. Regarding the second case, GSUs forward the *Req\_Pkts* to RSUs, and then RSUs deliver them to UAVs. Otherwise, they forward to nearby UAVs. The GCS forms a logical graph based on the location it administrates, including both GSU and UAV. Afterward, GCSs forward information to MC. Based on the information received from different GCSs, MC builds a global topology graph that includes all the areas covered.

2) *Distribution Phase*: The distribution of requested content within the aerial and ground layer depends on caching status of the desired content. If the requested content is cached within the same domain, forwarding the *Req\_Pkt* to GCS returns the content with the appropriate signature from the nearest node. However, for fresh or non-cached content, the consumer GSU checks whether UAV or GCS is the neighboring or next-hop node. If GCS is not the next-hop node, then GSU sends the *Req\_Pkt* to UAV first, and then UAV forwards to GCS. GCS makes an entry for the request and checks whether the entry exists or not in the domain that it maintains. If the content does not exist, it forwards the *Req\_Pkt* to MC and the *Res\_Pkt* to UAV. In this scenario, the *Req\_Pkt* carries an ACK entry that tells the UAV to wait for a certain amount of time. Now that

TABLE II  
NOTATIONS USED IN THIS WORK

Symbol	Representation
$M$	Number of UAVs
$\mathcal{U}$	Set of UAVs
$N$	Number of GSUs
$\mathcal{G}$	Set of GSUs
$x, y, z$	Aerial and ground units
$p$	Request-driven traffic
$r$	Response-driven traffic
$\delta$	Service processing capacity
$\Delta$	Set of service processing capacity
$\lambda$	Request for a service (or content)
$\Lambda$	Set of service requests
$\nu$	Response (service retrieval) rate for $\lambda$
$\Upsilon$	Total response rate for $\lambda$
$Q$	Flow entries waiting to be installed in queue
$\mathcal{D}$	Average delay
$a$	Contents in the network
$f_{xy}^{(a)}$	Content forwarded from one node to another
$\tau^{(p)}$	Threshold defined by requesting node
$\mathbf{A}$	Adjacency Matrix between UAVs
$\mathbf{B}$	Adjacency Matrix between UAV and GCU units
$\mathbf{F}$	Forwarding matrix
$\mathbf{E}, \mathbf{C}$	Auxiliary matrix is utilized as a form of heuristics to define function for GCS and MC, respectively
$\mathbb{B}^K, \mathbb{B}^J$	Bipartite graphs with service requests ( $K$ ) and available computing slots ( $J$ )

MC receives a *Req\_Pkt* from GCS, it immediately searches for its global topology entries and looks for the remote node with the content. If MC cannot locate the node, it prepares a *Res\_Pkt* with a NACK, sends it back to GCS, which finally arrives at GCU to discard the request. Suppose the requested content is found by MC, which may be located in a different domain, then it immediately looks for the best path to reach the origin of the content, and it forwards a *Req\_Pkt* to those nodes located in that discovered path. After receiving the *Req\_Pkt* from MC, GCSs install flows in their respective UAVs, allowing the content to be delivered as a data packet from the remote location to the requested GSUs using hop-by-hop communication.

### C. System Model

We use  $M$  and  $N$  to define the number of UAVs and ground sensing unit (GSU) operating in the aerial and ground layer, respectively. Table II lists the notations used in this work. Note that, we use content and service interchangeably. The main variables and their definitions from the perspective of system model are detailed below.

- $\mathcal{U} = \{u_1, u_2, u_3, \dots, u_M\}$  and  $\mathcal{G} = \{g_1, g_2, g_3, \dots, g_N\}$  are used to represent a set of UAVs and GSUs operating in aerial and ground layer, respectively;
- $\Delta^{(r)} = \{\delta_1^{(r)}, \delta_2^{(r)}, \dots, \delta_M^{(r)}, \delta_{M+1}^{(r)}\}$  denotes set of computational service processing capacity in terms of bandwidth and CPU for each UAV  $u_x \in \mathcal{U}$  and for each GCS (defined by  $\delta_{M+1}^{(r)}$ );
- $\Lambda = \{\lambda_1, \lambda_2, \dots, \lambda_M\}$  is set of services requested by UAV to GCS  $g_y \in \mathcal{G}$ , where  $\lambda_x = \lambda_x^{(p)} + \lambda_x^{(r)}$  contains both generic and computational types;

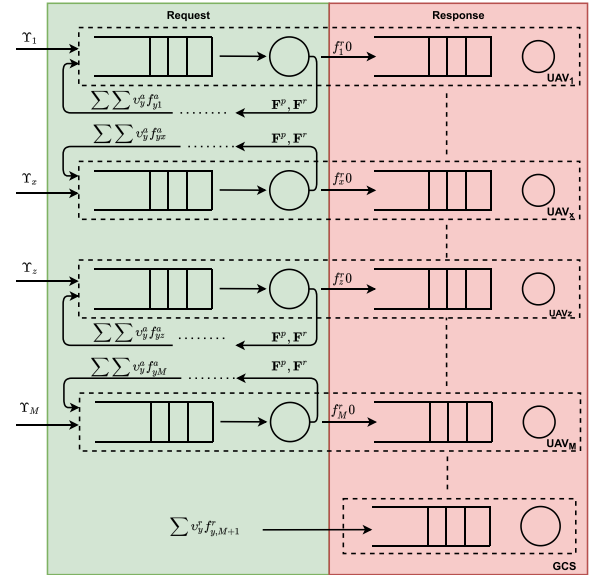


Fig. 5. Queue allocation procedure.

- $\mathbf{A}_{M \times (M+1)}$  is denoted as UAV adjacency matrix where  $a_{x,y} \rightarrow \{0, 1\}$  specifies an active connection between multiple UAVs  $u_x, u_y \in \mathcal{U}$  with multiple index such as  $M + 1, M + 2$ , which depict GCS and MC, respectively;
- $\mathbf{B}_{M \times N}$  symbolizes an UAV-GSU adjacency matrix where  $b_{y,z} \rightarrow \{0, 1\}$  specifies an active connection between UAV  $u_{ij} \in \mathcal{U}$  and GSU  $g_z \in \mathcal{G}$ .

Due to the mobility challenges in FANET such as three dimensional movement of UAVs along with line-of-sight (LoS) communication between aerial and ground units, the quality of link is often compromised [50]. Moreover, the traffic types in a traditional IC-SDN paradigm are vastly diversified based on the system's processing capability and the QoS requirements. The QoS has a critical role in UAV-based system due to the traffic processing orchestration initialized through the SDN controller in the networked system. By acknowledging these facts, we separate traffic types according to the information processing complexity of UAVs and GSUs in the network which is discussed in details in the next subsection.

### D. Queue Allocation Model

In the initial stage,  $\mathbf{A}_{M \times (M+1)}$  and  $\mathbf{B}_{M \times N}$  matrices are computed by GCS based on the location of GSUs and the availability of UAVs. We model the proposed FANET system based on *Open Jackson Network* [51], where each UAV is assigned with two continuous M/M/1 queues. In Fig. 5, we show the network queues on the left for both request and response-driven traffic types, which can be either generic or computational. On the right, we illustrate the queues for response-driven traffic that is computation-sensitive. In addition, we identify traffic patterns that are more suited to UAV-assisted networks because FANET, unlike VANET or MANET, is more delay-sensitive due to the dynamic speed of UAVs while operating under different conditions. Therefore, it is essential to identify different types of traffic based on the service availability and the computation

capability of the UAV. We use  $p$  to indicate generic traffic that is processed between aerial-ground units and GCS which follows the request-driven pull-based approach where the outcome of the traffic is influenced by the status of the source node that forwards request to upper layer. Conversely,  $r$  is used to present response-driven traffic that follows the push-based principle. Note that  $r$  is also computational traffic that relies upon the communication between UAV and GCS. We assume the information-centric softwareized FANET as an undirected graph where edges and nodes are presented as adjacency matrices. The pre-computation tasks are initiated by matrices  $\mathbf{A}_{M \times (M+1)}$  and  $\mathbf{B}_{M \times N}$  based on the location availability of the GSUs in the data plane layer. Both  $p$  and  $r$  traffic classes are used in processing the queue optimization for both UAVs, GSUs, GCSs and MC. Moreover, the information-centric forwarding policies managed by the softwareized controller determines the possible outcome of the content or services queued in forwarding nodes in aerial-ground layer. We use  $\lambda$  to define the request initialized by ground and aerial units through  $Req\_Pkt$  for a specific service or content to the nearest fixed data collecting point on the ground or the edge controller, i.e., GCS in our scenario. Furthermore,  $v$  determines the response rate for generic service requests  $\lambda$  that traverse back to the requested nodes as a result of successful content or service retrieval. Additionally,  $Q$  is used to denote the flow entries that are waiting to be installed in a queue on the UAVs.

The flow rules determines the forwarding policies as well as the output-queue. We model forwarding policies  $\mathbf{F}_{M \times (M+2)}^{(p)}$  and  $\mathbf{F}_{N \times (N+2)}^{(r)}$  for  $p$  and  $r$  traffic patterns. Due to the fact that, content  $a$  is processed within  $p$  or  $r$  traffic classes, we use  $f_{x,y}^{(a)}$  to denote the ratio of packet GCU  $g_y$  forwards to UAV  $u_x$  for traffic  $a \in \{p, r\}$ . Let  $f_{x,0}^{(p)}$  and  $f_{x,0}^{(r)}$  define request and response-driven packets (respectively) that are initially processed within the GSUs operating in ground layer at  $g_x$ , and forwarded to aerial layer to be processed by UAVs at  $u_x$ , and then finally departed from the data layer (aerial-ground) to be processed further in control layer. In the same way,  $f_{x,(M+1)}^{(r)}$  denotes the response-driven computation-sensitive packets that is forwarded to MC from sub-control edge layer to orchestrate flow installation instructions from the main controller in the cloud. The overall response rate  $v_x$  for GCU  $g_x$  is defined as  $v_x = v_x^{(p)} + v_x^{(r)}$ .

For both GCU and UAV that request for either a specific service or a content, we assume that they request with an average of  $1/\lambda$ , where Markov queue is used to achieve upper-bound acknowledgement time in the proposed model. For request and response traffic ( $\forall a \in \{p, r\}$ ), We define  $v_x^{(a)}$  as

$$v_x^{(a)} = \Upsilon_x^{(a)} + \sum_{y=1}^M f_{y,x}^{(a)} \cdot v_y^{(a)}, \quad (1)$$

where  $\Upsilon_x^{(a)} = \sum_{z=1}^N b_{x,z} \cdot \lambda_z^{(a)}$  is the response ratio for requested content initiated by  $z^{th}$  GCU unit. Likewise,  $\Upsilon_x^{(a)}$  is the total response rate for all the GCU units that are connected to UAV  $u_x$ .

Applying the Little's Law [52], the average delay for processing response-driven contents is calculated as

$$\mathcal{D}^{(r)} = \frac{\sum_{x=1}^M Q_x^{(p)} + \sum_{x=1}^{M+1} Q_x^{(r)}}{\sum_{z=1}^N \lambda_z^{(r)}}, \quad (2)$$

where  $Q_x^{(p)}$  indicates queued packets for contents that are waiting to be installed for generic traffic and  $Q_x^{(r)}$  denotes queued packets for response-driven computational traffic for UAV  $u_x$ . Let  $\Delta^{(p)} = \{\delta_1^{(p)}, \delta_2^{(p)}, \dots, \delta_N^{(p)}\}$  be the generic content processing rate for each UAV  $u_x \in \mathcal{U}$  where  $\lambda_x^{(p)}$  is measured in Mbps.

Therefore, we formulate the average response-driven traffic in the first queue as

$$Q_x^{(p)} = \frac{v_x^{(r)}}{\lambda_x^{(p)} - v_x}. \quad (3)$$

Similarly, we express the second queue value as

$$Q_x^{(r)} = \frac{v_x^{(r)} f_{x,0}^{(r)}}{\delta_x^{(r)} - v_x^{(r)} f_{x,0}^{(r)}}. \quad (4)$$

From the perspective of first and second queue we can rewrite  $\mathcal{D}^{(r)}$  as

$$\mathcal{D}^{(r)} = \frac{\sum_{x=1}^M \frac{v_x^{(r)}}{\delta_x^{(p)} - v_x} + \sum_{x=1}^{M+1} \frac{v_x^{(r)} \cdot f_{x,0}^{(r)}}{\delta_x^{(r)} - v_x^{(r)} \cdot f_{x,0}^{(r)}}}{\sum_{z=1}^N \lambda_z^{(r)}}. \quad (5)$$

Likewise, we calculate the average delay time for request-driven traffic as

$$\mathcal{D}^{(p)} = \frac{\sum_{x=1}^M \frac{v_x^{(p)}}{\delta_x^{(p)} - v_x}}{\sum_{z=1}^N \lambda_z^{(p)}}. \quad (6)$$

We assume the MC located in the cloud has the complete awareness of all content (or service) request  $\Lambda$ . Moreover, MC is also aware of the processing capacity  $\Delta^{(p)}$  for each participating node in the system grouped under generic traffic type. Besides, the link status of adjacency matrices  $\mathbf{B}_{M \times N}$  and  $\mathbf{A}_{M \times (M+1)}$  are also learned by MC.

### E. Problem Formulation

We present the joint forwarding and flow scheduling problem as

$$\text{minimize}_{\mathbf{F}^{(a)}} \mathcal{D}^{(r)} \quad (7)$$

$$\text{subject to } \sum_{u_x \in \mathcal{U}} b_{x,y} = 1 \quad \forall g_y \in \mathcal{G}, \quad (8)$$

$$v_x < \delta_x^{(p)} \quad \forall u_x \in \mathcal{U}, \quad (9)$$

$$v_x^{(r)} \cdot f_{x,0}^{(r)} < \delta_x^{(r)} \quad \forall u_x \in \mathcal{U}, \quad (10)$$

$$\sum_{y=0}^{M+1} f_{x,y}^{(a)} = 1 \quad \forall u_x \in \mathcal{U}, a \in \{p, r\}, \quad (11)$$

$$f_{x,y}^{(a)}, v_x^{(a)}, \delta_x^{(a)}, \lambda_x^{(a)} \geq 0, \quad (12)$$

$$\forall u_x \in \mathcal{U}, a \in \{p, r\}, y \in [0, M+1]$$

$$\mathcal{D}^{(p)} \leq \tau^{(p)} \quad (13)$$

Regarding the optimization problem formulated in (7), our primary objective is to minimize service provisioning delay for UAVs, acting as the primary content distributor to GSUs in the system. It satisfies the following *constraints*: (8) ensures the inter-communication between a single UAV and a single GСУ; (9) and (10) verify the validity of the queue allocation model; (11) and (12) specify that, each of the forwarded contents leaving one request queue will flow to other request or response-oriented queue within the active communication between UAV and GСУ; finally (13) guarantees that the information processing time for request-driven traffic  $p$  is not surpassed the system-defined threshold value  $\tau^{(p)}$ .

The joint forwarding and flow scheduling problem is NP-hard. Therefore, we divide the optimization problem into two parts based on [53]. First, we calculate the forwarding paths for traffic class  $p$  that travels through GCS. Then, we compute the forwarding paths for response-driven traffic class  $r$  between UAV, GСУ, GCS and MC based on the estimated flow allocation and traffic congestion. The first optimization solution – *Edge-Cloud Generic Flow Assignment (EGFA)* produces the entries from  $\mathbf{F}^{(p)}$  matrix as output which are then received as input by the *Edge-Cloud Computational Flow Assignment (ECFA)* module generating the final outcome as  $\mathbf{F}^{(r)}$  matrix.

#### IV. INFORMATION-CENTRIC SOFTWAREZED FLOW ASSIGNMENT SOLUTION

We solve the above-discussed optimization problem into two stages. Firstly, we propose a generic flow assignment method on the GCS to orchestrate the normal communication between the aerial-ground units, GCS and MC. After that, we present a computational flow assignment solution that utilizes both GCS and MC to allocate resources from the resource-rich cloud servers to the desired nodes in aerial and ground layer.

##### A. Edge-Cloud Generic Flow Assignment (EGFA)

The Edge-Cloud Generic Flow Assignment (EGFA) algorithm checks the initial entries from  $f_{x,y}^{(p)}$ , where  $x, y \leq M$  denote the flow entries for  $p$  traffics forwarded to GCS. Due to the complexity of the initial problem, the constraint (13) is relaxed to determine the PIT entries minimizing the service provision delay  $\mathcal{D}^{(p)}$ .

*Definition 1:* Given the set of available GSUs  $\mathcal{G}$ , set of operating UAVs  $\mathcal{U}$ , UAV adjacency matrix  $\mathbf{A}_{M \times M}$ , active aerial-ground connection matrix  $\mathbf{B}_{M \times N}$  and generic service requesting set  $\Lambda$ , the aim is to calculate the request-driven forwarding matrix  $\mathbf{F}_{M \times (M+2)}^{(p)}$  such that the average service provisioning delay for  $p$ -type traffic  $\mathcal{D}^{(p)}$  is minimized.

We make additional changes to the existing Dijkstra algorithm to guide the GCS to find the shortest path over the acyclic edge-weighted graph. In general, the Dijkstra algorithm calculates (among all the available paths) the shortest path of single or multiple sources to a single destination point. To serve this particular action, we assign cost function to UAV  $u_x(\zeta^{(p)}(u_x))$

and GСУ  $g_x(\zeta^{(p)}(g_x))$  as the intermediary PIT entries forwarded to GCS. We compute the cost function for UAV and GСУ as

$$\zeta^{(p)}(u_x) = \frac{1}{\delta_x^{(p)} - v_x^{(p)}} + \sum_{1 \leq y \leq (M+2)} e_{x,y} \cdot f_{x,y}^{(p)}, \text{ and} \quad (14)$$

$$\zeta^{(n)}(g_x) = \frac{1}{\delta_x^{(p)} - v_x^{(p)}} + \sum_{1 \leq y \leq (N+2)} e_{x,y} \cdot f_{x,y}^{(p)}. \quad (15)$$

In (14) and (15), the first fraction denotes the average delay for processing queued flow entries at UAV  $u_x$  and GСУ  $g_x$ , and the second one represents the average delay for processing queued path entries to GCS and MC.

Note that there are several reasons for the classical Dijkstra algorithm not to be compatible with our problem. Firstly, the link weight of the infrastructure-less FANET nodes is not stable because of continuous changes over time as nodes may vary when added to the forwarding path belonging to GCS's specified flow entries. Secondly, participating nodes in IC-SDN FANET have multiple paths when they are orchestrated by the control plane to be routed through one source to another destination. To tackle this challenge, we propose Algorithm 1, which establishes the shortest paths from the topology. Besides, we introduce two auxiliary matrices in the algorithm.

- The matrix  $\mathbf{E}$  monitors the cost of flows in the network that contains the pointer from a single node to its edge, for example,  $(e_{x,y} \in \mathbf{E}) > 0$  if  $u_y$  becomes the parent for  $u_x$  in an ordered tree while  $u_y$  defines the cost of path to GCS.
- The matrix  $\mathbf{C}$  monitors the requested flows for specific services that contains the reverse pointers from a single node to its subnodes. For example,  $c_{x,y} = 1$  if  $u_y$  becomes the child of  $u_x$ , otherwise  $c_{x,y} = 0$ .

At the start of the Algorithm 1,  $\mathbf{E}$  and  $\mathbf{C}$  matrices are initialized as starting point in line 2 and line 3 respectively, to define a parent connection from the GCS to the child node i.e., next hop node with the index  $M+2$  where the path cost is set to zero (in line 5). In line 6, the algorithm adds the one with the minimum path cost to GCS in the solution set by performing a greedy selection. Afterwards,  $v_x^{(p)}$  and matrices  $\mathbf{E}$ ,  $\mathbf{C}$  are updated in lines 8–10 and lines 26–28, respectively. In the end, the request-driven forwarding matrix  $\mathbf{F}^{(p)}$  is updated to adjust the flow-based PIT entries allowing the traffic to all the necessary paths and equalizing the values in  $(f_{x,y}^{(p)} \cdot \zeta^{(p)}(u_y))$ ,  $\forall y : e_{x,y} \geq 0$  to minimize the path cost in line 28. Here, we bypass the checking for space reason if  $f_{x,y}^{(p)} > 1$  in cases where we cannot balance the cost over all the parents.

In Fig. 6, we present a specific case with three main functions – `updateResponseRate`, `updateParentCost`, and `updateChildrenList` used in Algorithm 1. We denote  $u_2$  and  $u_1$  as active and non-active node which are non-explored in the initial stage. In the `updateResponseRate` stage, the flows travel from leaf-based nodes (UAVs and GSUs) to root-based nodes (GCS and MC), updating all the  $v_x$  in accordance with the connection to the tree. Afterwards, `updateParentCost` function updates each communication links' costs ( $e_{M+1, M+2} = 0$ ) starting from the communication between the next hop (in the path) and GCS to the nodes in the leaves. In the end, forwarding values  $f^{(p)}$  are



**Algorithm 1:** Edge-Cloud Generic Flow Assignment.

```

1 EGFA ( $\mathcal{U}, \mathbf{A}_{M \times M}, \mathbf{B}_{M \times N}, \Lambda, \Delta$ )
2 init:  $\mathbf{E}_{M+1 \times M+2}$  with  $e_{x,y} = -1$   $\{e_{x,y} \geq 0$  is path
   cost of  $y$  being parent to  $x\}$ 
3 init:  $\mathbf{C}_{M+1 \times M+2}$  with  $c_{x,y} = 0$   $\{c_{x,y} = 1$  is path cost
   of  $y$  being child to  $x\}$ 
4 init:  $\delta_x = false, v_x^{(p)} = Y_x^{(p)}, f_{x,y}^{(p)} = 0, 1 \leq x, y \leq M$ 
5 set  $e_{M+1, M+2} = 0$   $\{M+1$  is GCS and  $M+2$  is MC $\}$ 
6 while  $\exists u_z$  s.t.  $(\delta_z = false) \wedge (\exists y$  s.t.
    $e_{z,y} \geq 0) \wedge (\arg \min_z \varsigma^{(p)}(u_z))$  do
7    $\delta_z = true$ 
8 call updateResponseRate ( $u_{M+1}$ )
9 call updateParentCost ( $M+2, u_{M+1}, 0$ )
10 call updateChildrenList ( $u_z$ )
11 return  $F^{(p)}$ 
12 Function updateResponseRate ( $u_z$ )
13    $v_z \leftarrow Y_z$ 
14   forall  $u_x \ni c_{z,x} = 1$  do
15     if  $\delta_x = true$  then
16        $v_z \leftarrow v_z + (f_{x,z}^{(p)} \cdot \text{updateResponseRate}(u_x))$ 
17   return  $v_z$ 
18 Function updateParentCost ( $y, u_z, \text{full cost}$ )
19    $f_{z,y} \leftarrow \text{full-cost}$ 
20   forall  $u_x \ni c_{z,x} = 1$  do
21      $\text{updateParentCost}(z, u_x, \varsigma^{(p)}(u_z))$ 
22 Function updateChildrenList ( $u_z$ )
23   forall  $u_x$  and  $b_{z,x} = 1$  with  $b_{z,x} \in \mathbf{B}$  do
24     if  $\delta_x = false$  then
25        $c_{z,x} = 1$ 
26   forall  $u_x \ni c_{z,x} = 1$  do
27      $e_{z,x} \leftarrow \varsigma^{(p)}(u_z)$ 
28      $f_{x,y}^{(p)} \leftarrow \frac{\prod_{(n:e_{x,n} \geq 0, n \neq y)} \varsigma^{(p)}(u_n)}{\sum_{(n:e_{x,n} \geq 0)} \prod_{(d:e_{x,d} \geq 0, d \neq n)} \varsigma^{(p)}(u_d)}, \forall y :$ 
      $e_{x,y} \geq 0$ 
    
```

updated by the updateChildrenList function within the node  $u_2$ . Note that, the proposed algorithm's output refers to a destination-based directed acyclic graph (denoted as  $\mathbf{F}^{(p)}$  matrix) and is applied to GCS due to the fact that, each nodes have the option to choose from multiple paths to reach their destination.

### B. Edge-Cloud Computational Flow Assignment (ECFA)

Now that, we have already discussed on the information-centric flow optimization in the edge-layer, it is necessary to optimize the traffic flow between edge and cloud layers. Here, the ECFA algorithm allocates services from computational resource-oriented nodes located in the cloud (i.e., core network) to the edge nodes. Based on the previously discussed system and queue allocation model, it is essential to select the best forwarding path from source to destination node for  $r$  traffics. To solve this issue, we present the path selection module as a bipartite weighted graph. We denote i)  $\mathbb{B}^K = \{b_1^K, b_2^K, \dots\}$  as set of service requests for pending computational tasks with  $|\mathbb{B}^K| =$

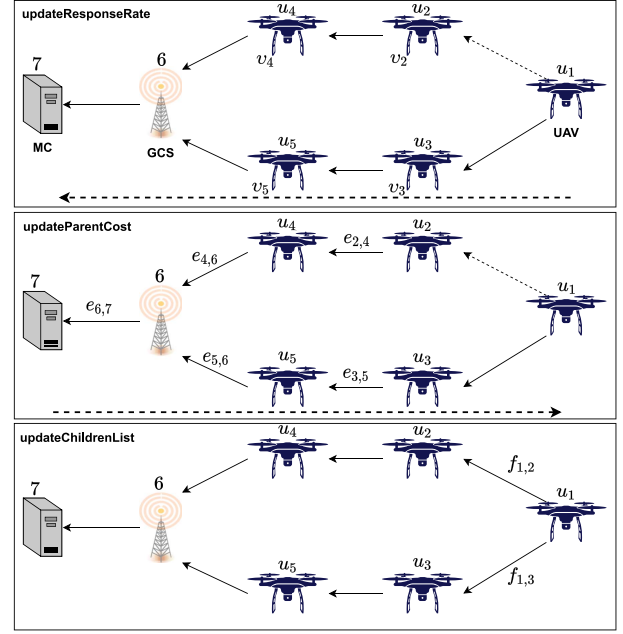


Fig. 6. Schematic execution overview of Algo. 1.

$\sum_{\lambda_x \in \Lambda} \lambda_x^{(r)}$ , and ii)  $\mathbb{B}^J = \{b_1^J, b_2^J, \dots\}$  as set of available computing slots on the cloud layer ( $|\mathbb{B}^J| = \sum_{\delta_x^{(r)} \in \Delta^{(r)}} \delta_x^{(r)}$ ). Both set of service requests and available computing slots are measured per time unit. We assign  $\psi : \mathbb{B}^K \times \mathbb{B}^J \rightarrow \mathbb{I}$  as the weight function that represents the incentives of service request allocation  $\mathbb{B}^K$  to a computing slot in  $\mathbb{B}^J$ . Therefore, an asymmetric assignment problem is considered to show that the available computing slots are strictly greater than the requests for services to perform computing tasks which also satisfies (10)'s requirements. Let  $\mathbb{R}(b_z^K) = u_x$  be the mapping function that provides to UAV with the request  $b_z^K$ ; consequently, let  $\mathbb{P}(b_x^J) = u_y$  be the function that provides to UAV with the computation slot  $b_x^J$ . The assignment problem aims to find out the optimal assignment set  $\mathcal{A} = \{(b_x^K, b_y^J) : b_x^K \in \mathbb{B}^K, b_y^J \in \mathbb{B}^J\}$  to maximize the total incentives  $\sum_{(b_x^K, b_y^J) \in \mathcal{A}} \psi(b_x^K, b_y^J)$  depending on the constraints that ensure each of the requests are allocated within a single slot to accommodate no more than one computed request.

The solution presented in Algorithm 2 improves the fundamental forward/reverse auction method if the weights change dynamically with the time. Note that, each assignment leads to modifications in the  $\psi$  function because  $v_x^{(r)}$  changes due to UAVs located on the selected path (Algorithm 2: Line 10). Therefore, Algorithm 2 executes a series of forward/reverse iterations where the slot for request assignment is added or removed from the final outcomes within each iteration. We denote  $\mathbb{I}(b_x^K)$  as the index of calculated path for  $b_x^K$ . We model the incentive function  $\psi(b_x^K, b_y^J)$  of request assignment as follows:

$$\psi(b_x^K, b_y^J) = \frac{1}{1 + \varsigma^{(r)}(b_x^K, b_y^J)}, \quad (16)$$

where  $\varsigma^{(r)}(b_x^K, b_y^J)$  is the cost function for delay regarding the traffic type  $r$  from UAV  $\mathbb{R}(b_x^K)$  to  $\mathbb{P}(b_y^J)$ . Here, we calculate the cost by applying Dijkstra ( $\mathbb{R}(b_x^K), \mathbb{P}(b_y^J)$ ) to find out the shortest

---

**Algorithm 2:** Edge-Cloud Computational Flow Assignment.

---

```

1 ECFA  $\mathcal{U}, \mathbf{A}_{M \times M}, \mathbf{B}_{M \times N}, \Lambda, \Delta^{(p)}, \Delta^{(r)}, \mathbb{B}^K, \mathbb{B}^J$ 
2 forall  $u_x \in \mathcal{U}$  do
3   if  $\Upsilon_x^{(p)} > 0$  then
4     calculate Dijkstra ( $u_x, GCS$ )
5     update path with  $f_{y,z}^{(p)}, \forall u_z \in \mathcal{U}$  and  $\forall u_y$ 
6     update path with  $\lambda_y^{(p)}, \forall u_y$ 
7    $\mathbb{I}(b_x^K) \leftarrow \{\}, \forall b_x^K \in \mathbb{B}^K$ 
8 while assignments for  $\forall b_x^K \in \mathbb{B}^K$  are not available in
    $\mathcal{A}$  do
9   calculate Dijkstra ( $u_x, u_y$ ),  $\forall u_x, u_y \in \mathcal{U}$ 
10  update  $\psi(b_x^K, b_y^J), \forall b_x^K \in \mathbb{B}^K, b_y^J \in \mathbb{B}^J$  using
    Dijkstra ( $\mathbb{R}(b_x^K), \mathbb{P}(b_y^J)$ )
11  execute auction algorithm with single forward or
    reverse step
12  if assignment  $\langle b_x^K, b_y^J \rangle$  is new then
13     $\mathbb{I}(b_x^K) \leftarrow \text{Dijkstra}(\mathbb{R}(b_x^K), \mathbb{P}(b_y^J))$ 
14  else if remove assignment  $\langle b_x^K, b_y^J \rangle$  then
15     $\mathbb{I}(b_x^K) \leftarrow \{\}$ 
16  update  $v_x^{(r)}$  and  $f_{x,y}^{(r)}$  based on path index
     $\mathbb{I}(b_x^K), \forall b_x^K \in \mathbb{B}^K$ 

```

---

path with edge ( $u_x \rightarrow u_y$ ) weight which is indicated as:  $\frac{1}{\delta_x^{(p)} - v_x}$ . More specifically, we use  $\mathbb{I}(b_x^K) = \{\mathbb{R}(b_x^K), \dots, \mathbb{P}(b_y^J)\}$  to be the path utilized to arrive at  $\mathbb{P}(b_y^J)$ . Therefore, the cost function becomes:

$$\zeta^{(r)}(b_x^K, b_y^J) = \left( \sum_{u_z \in \mathbb{I}(b_x^K)} \frac{1}{\delta_z^{(p)} - v_z} \right) + \frac{1}{\delta_{\mathbb{P}(b_y^J)}^{(r)} - \left( v_{\mathbb{P}(b_y^J)}^{(r)} \cdot f_{\mathbb{P}(b_y^J),0}^{(r)} \right)}. \quad (17)$$

### C. Computational Complexities of Proposed Algorithms

Here we present the computational complexity for both EGFA and ECFA algorithm. Note that, both algorithms are based on Dijkstra algorithm that has the complexity of  $\mathcal{O}(N^2)$  in its basic form. This complexity is shown in Alg. 1 (line 6) where the *argmin* operator is  $\mathcal{O}(N)$  and the primary *while* loop is executed  $N$  times. Besides, the functions *updateResponseRate* and *updateParent* dominate the computation process in that loop as they travel the entire graph to update the response rates with their costs. Therefore, EGFA algorithm's computation complexity is  $\mathcal{O}(N^3)$ . On the other hand, we use Alg. 2 to solve with asymmetric assignment problem using weighted bipartite graph with two asymmetric sets: i) service requests sets with cardinality  $|\mathbb{B}^K|$  and, ii) available computing slots with cardinality  $|\mathbb{B}^J|$ . We use auction algorithm to solve the asymmetric assignment problem in  $\mathcal{O}(|\mathbb{B}^K| |\mathbb{B}^J| \cdot \log(n))$ , where we denote  $n$  as a parametric value [54]. Note that, an additional execution time has

TABLE III  
SPECIFICATION OF SIMULATION PARAMETERS

Parameter	Value
Number of UAVs	10 to 500
Link between UAVs	8 Mb/sec.
Link between UAVs and GSU	30 Mb/sec.
Link between GSUs and GCSs	100 Mb/sec.
Altitude between UAVs and GSUs	100 m
Payload size	2000 KB
Request rate	50 packets/sec.
Simulation runs	20
Simulation duration	600 sec.

been added in our implementation so that function  $\psi(b_x^K, b_y^J)$  can update the the cost matrix (Alg.2, line 4 and 5). By adding the complexity  $\mathcal{O}(N^2)$  from Dijkstra's algorithm and complexity  $\mathcal{O}(|\mathbb{B}^K| |\mathbb{B}^J|)$  from matrix update, the total computation complexity becomes  $\mathcal{O}(|\mathbb{B}^K| |\mathbb{B}^J| \cdot \log(n) \cdot (|\mathbb{B}^K| |\mathbb{B}^J| + N^2))$ .

## V. PERFORMANCE EVALUATION

This section evaluates the efficiency and efficacy of our proposed solution and compares it against state-of-the-art solutions. We start by presenting a technical overview of the testbed, including the hardware specification, applications used, and the experimental parameters. Then, we present detailed insights regarding each experiment alongside the evaluation results.

### A. Testbed and Simulation Methodology

Developing a softwarized UAV-based testbed on large scale is challenging in terms of the availability of compatible hardware suited to implement SDN controllers and ICN enabled nodes. Therefore, we deploy a simulated testbed that is flexible enough to initialize both SDN and ICN modules. We use a similar system with i5-9400F 2.9 GHz processor, 16 GB DDR4 memory, 6 GB DDR5 additional memory for GPU, and NVMe m.2 SSD interface storage for running the simulated experiments. Similar setup was used to evaluate controller performances for the work in [55]. The technical specification of the testbed environment is presented in Table III.

We use ns-3 [56], a discrete-event simulator that is specifically designed for networked system and has mobility support to run specifically MANET and VANET scenarios. To create information-centric communication between the participating nodes, we use ndnSIM [57], an additional module of ns-3 simulator. To compare our work against some of the existing prototypes, we choose UAVCO [19], and JOAR [37] as they focus on to SDN-based implementations that includes the flow computation, traffic optimization, and task assignment. The simulated area  $1500 \times 1500$  m. To initiate wireless communication between UAV nodes, we set a frequency bandwidth and link of 100 MHz and 1 GHz, respectively. The total simulation time 600 sec. Each experiment is initiated 20 times to avoid any possible errors and randomness in the simulated testbed. Therefore, the calculated average value is kept to a 95% confidence level for the proposed scheme's statistical analysis. In the experimental scenario, we deploy UAVs as consumers or request forwarders for other immediate sensor nodes. For example, a device on

the ground requests the UAV to find some desired content. In such an application scenario, the UAV has to provide support for communication to ground units (e.g., vehicles) that generate a request which immediately goes to UAVs for further processing in the aerial layer.

### B. Result Analysis and Discussion

In this section, we discuss the performance analysis of the proposed information-centric software-driven UAV framework and its comparison with existing software-driven UAV solutions. We categorize the analysis into four portions.

1) *Throughput Measurement*: In this experiment, we first study the impact on throughput of our system against the increasing number of UAVs. We then perform another throughput analysis by retaining the number of UAVs at 400 while increasing the number of unique contents in those fixed UAVs. In general, the consumer UAVs forward a maximum number of requests to GCS depending on the service type. Unlike latency measurement, where we count the destination node's ACK rate, the throughput measurement concentrates specifically on the number of service requests that the consumer UAVs can forward to the network without relying on the response packet from GCS that either be ACK or NACK.

a) *Increasing number of UAVs*: In this experiment, we increase the UAVs from 10 to 500, and initiate maximum of 50 requests per second. We calculate the average throughput. Fig. 7(a) compares the average throughput of the proposed solution against UAVCO and JOAR. In the initial phase, we start by 10 consumer UAVs simultaneously send service requests to GCS. For 10 UAVs, the proposed scheme achieves an average throughput of 8.3 Mbps which is 2.35 Mbps higher than UAVCO's 5.9 Mbps and 4.5 Mbps higher than JOAR's 3.75 Mbps. With the maximum number of 500 UAVs, our solution reaches 11.7 Mbps outperforming UAVCO and JOAR despite obtaining 10.8 Mbps and 9.7 Mbps, respectively.

b) *Increasing number of content*: In this experiment, the simulator instructs 400 UAVs from the aerial layer to send requests for an increasing number of unique contents to the control layer. Fig. 7(b) depicts the throughput result comparison of our solution and others. For 200 unique contents, we achieve an average throughput of 10.3 Mbps that is 2.2 Mbps better than UAVCO's 8.61 Mbps and 2.4 Mbps more than JOAR's 7.84 Mbps. The trend remains similar for all three solutions until 1200 unique content items. When the content number is increased further up to 2000, the proposed solution's throughput remains stable within 9.86 Mbps while UAVCO's performance drops to 7.88 Mbps and JOAR's throughput declines further to 6.72 Mbps. The proposed solution achieves stable throughput performance despite increasing unique contents due to ICN-based on-path caching implementation into aerial and ground units.

2) *Computational Load & Energy Consumption*: In this evaluation we discuss the analysis of the proposed system in terms of computation consumption on the simulated testbed.

a) *Computational load*: The increased amount of computation load determines the weight of the proposed architecture.

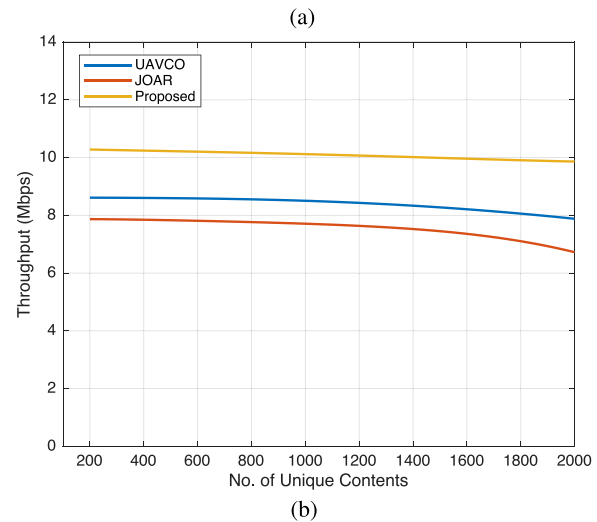
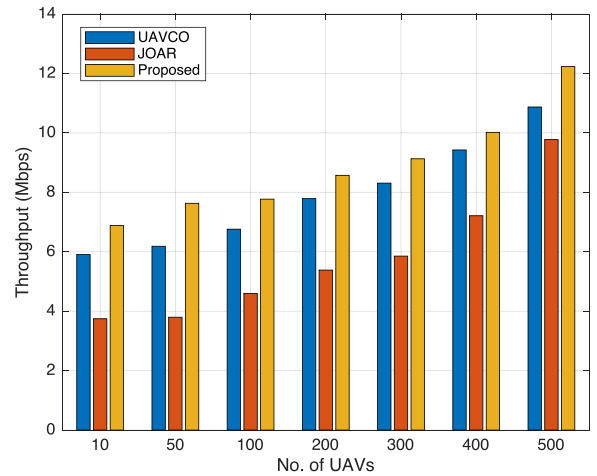


Fig. 7. Throughput Comparison: (a) Increasing UAVs as consumers, (b) Increasing unique content for 400 UAVs.

In general, the computational load evaluates the proposed algorithm's efficiency in terms of how much resources are being consumed to perform computation-sensitive tasks like service requests, queue processing, content discovery, and distribution. Firstly, we measure the available system load of the simulation testbed. Then, we execute the experiment to check the system load when UAVs retrieve the whole chunk of content from the destination node, from either GSU or another UAV located in a different domain. Then we increase the number of UAVs to 500 and check the system load for computation. Fig. 8(a) shows the computation load comparison of the proposed model against UAVCO and JOAR. With an increasing number of UAVs, the computation load for JOAR and UAVCO increased from 0.2% to 0.8% and 0.15% to 0.35%, respectively. In contrast, while running a similar experiment using our prototype, the computation load remains 0.11% to 0.29%, which is 0.51% and 0.06% lighter than JOAR and UAVCO, respectively.

b) *Energy consumption*: Energy utilization is one of the critical measurements to define the feasibility of the architecture. In this experiment, we consider cumulative energy consumption for tasks being transmitted against increasing consumer UAVs

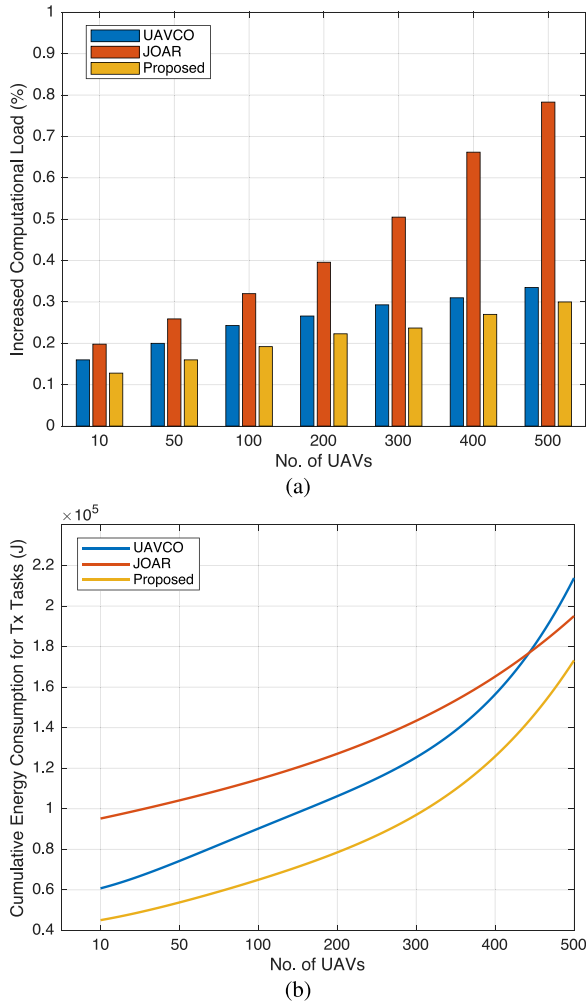


Fig. 8. Performance Comparison: (a) Computational load, (b) Energy consumption.

in the system. We consider a successful request forwarded by UAVs to GCS as transmitted tasks. The request may bring back ACK or NACK in return from the neighboring nodes, depending on the logical decision-making by the control layer. Fig. 8(b) shows the energy consumption as the UAVs are added into the system in all three scenarios. However, the proposed solution has consumed the least amount of energy as compared to others. The queue allocation process reduces the task execution time alongside selecting optimal paths orchestrated by GCS and MC. As a result, the consumer UAV consumes the least amount of energy for transmitted tasks despite UAVs increase in the system.

3) *End-to-End & Average Packet Delay*: The handover latency directly impacts the end-to-end delay between requests initiated by the consumer UAVs until received from the nearest UAV (in the aerial layer) or GSU (in the ground layer).

a) *End-to-end (E2E) delay*: To evaluate end-to-end delay for packets arriving at the consumer UAV from the producer UAV, we randomly distribute the UAVs in different areas and calculate the time between the request initiated by a consumer UAV until GSU serves it. Fig. 9(a) compares the end-to-end delay of the proposed solution against UAVCO and JOAR protocol. For 10 UAVs, the E2E delay remains at 8.2 ms, almost 2.9 ms

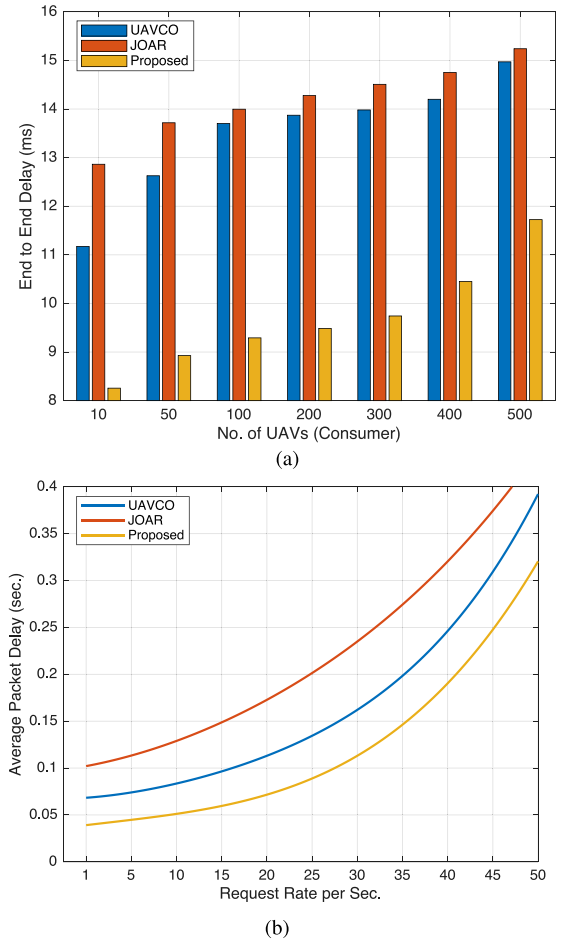


Fig. 9. Performance Comparison: (a) End-to-end delay, (b) Average packet delay.

and 4.6 ms less than UAVCO and JOAR that achieve 11.2 ms and 12.9 ms, respectively. For 100 consumer UAVs, although the E2E delay increases to 9.2 ms, the value remains 4.4 ms and 4.7 ms less than UAVCO and JOAR. When the maximum number of UAVs is increased to 500, the E2E delay increases to 11.7 ms while the value remains 3.2 ms and 3.5 ms less than UAVCO's 14.9 ms and JOAR's 15.2 ms.

b) *Average packet delay*: Fig. 9(b) depicts comparisons of average packet delay for 400 UAVs, generating 1 to 50 requests per second. The average delay of content or service requested by a consumer UAV denotes the average time interval between forwarding the request packet to the controller and receiving the data packet. The experiment result shows that the average delay of each softwarezied solution increases with the frequency of requests. For 10 requests, the average delay of the proposed solution is reduced to 0.03 s and 0.08 s compared to UAVCO and JOAR, respectively. Similarly, when 400 UAVs forward 50 requests, the average delay remains 0.07 s less than UAVCO and 0.12 s lesser than JOAR. Therefore, our proposed model has the lowest average packet delay compared to UAVCO and JOAR. The GCS and MC acknowledge the priority level of the requested packet coming from the aerial and ground layer, which helps to achieve low packet delay despite the number of increased requests.

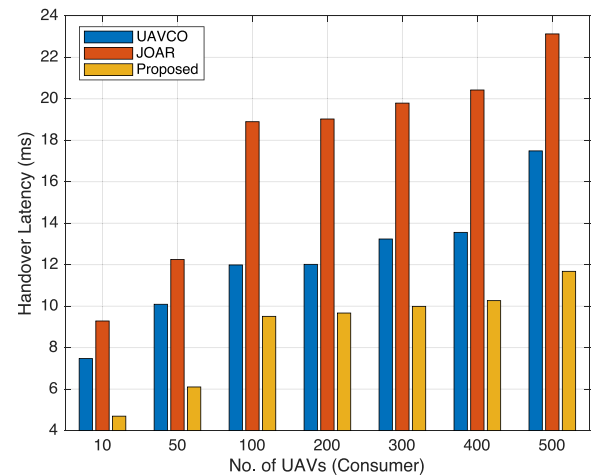
4) *Handover Latency, Packet Loss, & Link Utilization*: In this evaluation, we measure the latency, packet loss and link utilization efficiency between our proposed model and the existing software-driven UAV solutions.

a) *Handover latency*: The performance of handover latency is affected by an increasing number of UAVs initiating service requests at the same time. The results shown in Fig. 10(a) compares the handover latency performance of the proposed solution against UAVCO and JOAR protocol. For 10 consumer UAVs, the latency stays at 4.6 ms, almost 2.7 ms, and 4.5 ms less than UAVCO and JOAR that achieve 7.4 ms and 9.2 ms, respectively. For 200 consumer UAVs, despite the latency increases to 9.6 ms, the value remains 2.3 ms and 9.3 ms less than UAVCO and JOAR. When the maximum number of UAVs is increased to 500, the latency increases to 11.68 ms while the value remains 5.8 ms and 11.4 ms less than UAVCO's 17.48 ms and JOAR's 23.12 ms. The comparison result illustrates that using the proposed IC-SDN protocol improves the handover latency while the requested services are shifted within the nodes in aerial and ground layers.

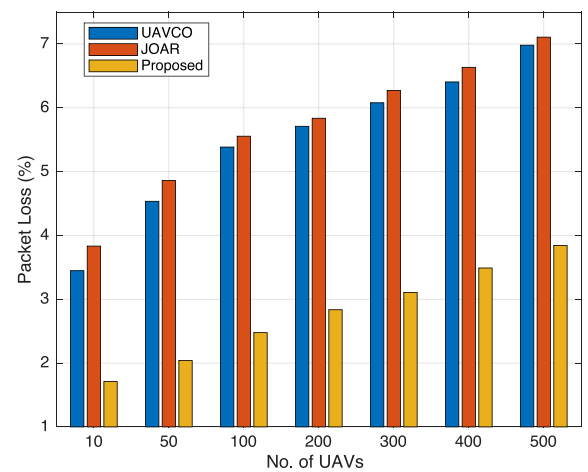
b) *Packet loss*: In general, the proposed solution improves the average number of packet losses by 2.7% and 2.9% compared to UAVCO and JOAR, respectively. Fig. 10(b) depicts the comparison between the number of consumer UAVs and dropped packets during the content discovery and distribution process. In the initial stage of the experiment with 10 UAVs, the packet loss stays within 1.7%, which is almost 1.8% and 2.2% less than UAVCO and JOAR, which outputs 3.4% and 3.9%. With 100 consumer UAVs, the packet loss delay increases slightly to 2.5%, which is 2.9% and 3.1% less than UAVCO's 5.4% and JOAR's 5.6%. The increasing number of 500 UAVs results in 3.8% of packet loss while 6.9% occurs in UAVCO protocol and 7.1% in JOAR protocol.

c) *Link utilization*: We compare the normalized link utilization of our model, UAVCO, and JOAR against request per second, as shown in Fig. 10(c). In this experiment, we keep UAVs and unique contents at 400 and 1000, respectively. The frequency of requests sent by those UAVs is increased to evaluate the link utilization in normalized outputs. In the initial stage, with a single request, the link utilization of UAVCO and JOAR results in 3% and 4%, respectively. However, the proposed framework achieves 20% normalized link utilization. As the number of requests is increased up to 50 requests per second, the proposed solution keeps a similar trend showcasing 98% utilization which is 3% less than UAVCO and 14% less than JOAR. Given the same input parameters, the overall link utilization of competing solutions reaches 100% at about 45 requests per second, after which no further increase in communication load is possible. However, in the proposed solution the link utilization is much less (for same input parameters), hence allowing more load to be placed on the networks, giving higher throughput. 100% link utilization is achieved at 55 requests per second.

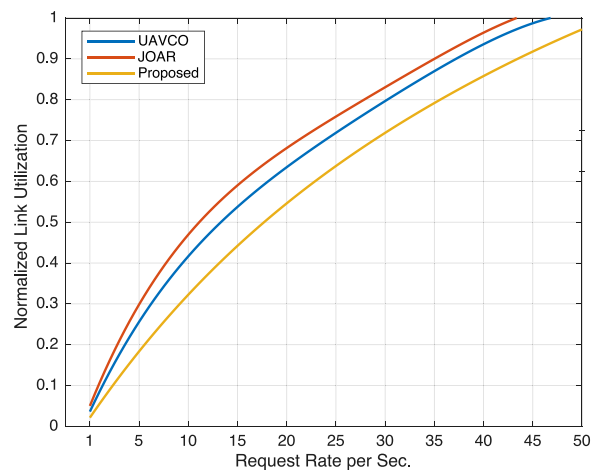
For both [19] and [37], service offloading and computation depend on UAVs' processing, computing capacity, and trajectories. For example, in [37], if a UAV fails to compute tasks within an assigned time due to its resource and energy constraints, it impacts the overall performance of the networked system, including end-to-end delay and handover latency and system



(a)



(b)



(c)

Fig. 10. Performance Comparison: (a) Handover latency, (b) Packet loss, and (c) Link utilization.

cost. On the other hand, [19] fails to address the multi-controller coordination issue and largely depends on the single controller located in the cloud to orchestrate most network provisioning tasks. As a result, centralized SDN controller-based solutions suffer poorly in computational load, throughput, and energy cost when the number of UAVs or other sensor nodes increases in the

network or a single point of failure occurs to the controllers. In contrast, we utilize ICN, which offers multiple advantages to UAV-based communication with its named-based traffic management and in-network caching facilities, which solves several issues. The central theme of ICN is that it assures that the content is always available regardless of how complex the network becomes. Each relay and forwarding node are essential to ensure the network does not rely on a single node, i.e., UAV or a specific layer. Therefore, if a UAV fails because of limited CPU or energy, it always forwards packets to nearby nodes to make the content available for the requesting node. We introduce multiple SDN controllers in the architecture because the single SDN control management system does not perform in the wireless environment as there are different types of nodes located in different layers. Therefore, the hierarchical distributed control layer distributes tasks to different controllers on the domain so that domain controllers do not need to carry an additional burden.

## VI. CONCLUSION

This paper presents an IC-SDN UAV-assisted system architecture where participating nodes from different layers hierarchically collaborate. The separation of distributed control units collects necessary information from UAVs and the cloud, which improves the solution's scalability. We studied queuing delay behavior of UAVs through traffic scheduling and forwarding contents via distributed controllers placed in the edge and cloud. The M/M/1 queue concept has been adopted to schedule flow entries in the PIT tables for UAV and GSU based on their geographical positioning and service availability. Experimental results show that the throughput, delay, energy consumption, and other nodal/network parameters are in favor of the proposed solution.

In future, we plan to improve the existing location-aware caching, computing and networking through hierarchical IC-SDN controllers to control the optimization of UAV swarms. Besides, we plan to shift some of the controller functions from the ground control system to specific UAVs to improve the controller-UAV interactions. Finally, machine learning techniques such as reinforcement learning can be applied to the control plane to improve the information-centric synchronization between the nodes, packet forwarding, and content caching.

## REFERENCES

- [1] S. Wijethilaka and M. Liyanage, "Survey on network slicing for Internet of Things realization in 5G networks," *IEEE Commun. Surv. Tut.*, vol. 23, no. 2, pp. 957–994, Apr.–Jun. 2021.
- [2] R. Borralho, A. Mohamed, A. Qudus, P. Vieira, and R. Tafazolli, "A survey on coverage enhancement in cellular networks: Challenges and solutions for future deployments," *IEEE Commun. Surv. Tut.*, vol. 23, no. 2, pp. 1302–1341, Apr.–Jun. 2021.
- [3] N.-N. Dao et al., "Survey on aerial radio access networks: Toward a comprehensive 6G access infrastructure," *IEEE Commun. Surv. Tut.*, vol. 23, no. 2, pp. 1193–1225, Apr.–Jun. 2021.
- [4] A. Baltaci, E. Dinc, M. Ozger, A. Alabbasi, C. Cavdar, and D. Schupke, "A survey of wireless networks for future aerial communications (FACOM)," *IEEE Commun. Surv. Tut.*, vol. 23, no. 4, pp. 2833–2884, Oct.–Dec. 2021.
- [5] M. Mozaffari, W. Saad, M. Bennis, Y.-H. Nam, and M. Debbah, "A tutorial on UAVs for wireless networks: Applications, challenges, and open problems," *IEEE Commun. Surv. Tut.*, vol. 21, no. 3, pp. 2334–2360, Jul.–Sep. 2019.
- [6] J. Xu, K. Ota, and M. Dong, "Big data on the fly: UAV-mounted mobile edge computing for disaster management," *IEEE Trans. Netw. Sci. Eng.*, vol. 7, no. 4, pp. 2620–2630, Oct.–Dec. 2020.
- [7] L. Gupta, R. Jain, and G. Vaszkun, "Survey of important issues in UAV communication networks," *IEEE Commun. Surv. Tut.*, vol. 18, no. 2, pp. 1123–1152, Apr.–Jun. 2016.
- [8] A. Fotouhi et al., "Survey on UAV cellular communications: Practical aspects, standardization advancements, regulation, and security challenges," *IEEE Commun. Surv. Tut.*, vol. 21, no. 4, pp. 3417–3442, Oct.–Dec. 2019.
- [9] H. Wang, H. Zhao, J. Zhang, D. Ma, J. Li, and J. Wei, "Survey on unmanned aerial vehicle networks: A cyber physical system perspective," *IEEE Commun. Surv. Tut.*, vol. 22, no. 2, pp. 1027–1070, Apr.–Jun. 2020.
- [10] A. Sharma et al., "Communication and networking technologies for UAVs: A survey," *J. Netw. Comput. Appl.*, vol. 168, 2020, Art. no. 102739.
- [11] D. S. Lakew, U. Sa'ad, N.-N. Dao, W. Na, and S. Cho, "Routing in flying ad hoc networks: A comprehensive survey," *IEEE Commun. Surv. Tut.*, vol. 22, no. 2, pp. 1071–1120, Apr.–Jun. 2020.
- [12] W. Zhang, L. Li, N. Zhang, T. Han, and S. Wang, "Air-ground integrated mobile edge networks: A survey," *IEEE Access*, vol. 8, pp. 125998–126018, 2020.
- [13] I. Alam et al., "A survey of network Virtualization techniques for Internet of Things using SDN and NFV," *ACM Comput. Surv.*, vol. 53, no. 2, pp. 35:1–35:40, Apr. 2020.
- [14] Z. Latif, K. Sharif, F. Li, M. M. Karim, S. Biswas, and Y. Wang, "A comprehensive survey of interface protocols for software defined networks," *J. Netw. Comput. Appl.*, vol. 156, Apr. 2020, Art. no. 102563.
- [15] N. Zhang, S. Zhang, P. Yang, O. Alhussain, W. Zhuang, and X. S. Shen, "Software defined space-air-ground integrated vehicular networks: Challenges and solutions," *IEEE Commun. Mag.*, vol. 55, no. 7, pp. 101–109, Jul. 2017.
- [16] O. Sami Oubbati, M. Atiquzzaman, T. Ahamed Ahanger, and A. Ibrahim, "Softwarization of UAV networks: A survey of applications and future trends," *IEEE Access*, vol. 8, pp. 98073–98125, 2020.
- [17] V. Hassija et al., "Fast, reliable, and secure drone communication: A comprehensive survey," *IEEE Commun. Surv. Tut.*, vol. 23, no. 4, pp. 2802–2832, Oct.–Dec. 2021.
- [18] O. S. Oubbati, M. Atiquzzaman, P. Lorenz, A. Baz, and H. Alhakami, "Search: An SDN-enabled approach for vehicle path-planning," *IEEE Trans. Veh. Technol.*, vol. 69, no. 12, pp. 14523–14536, Dec. 2020.
- [19] L. Zhao, K. Yang, Z. Tan, X. Li, S. Sharma, and Z. Liu, "A novel cost optimization strategy for SDN-enabled UAV-assisted vehicular computation offloading," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 6, pp. 3664–3674, Jun. 2021.
- [20] C. Zhang, M. Dong, and K. Ota, "Deploying SDN control in internet of UAVs: Q-learning-based edge scheduling," *IEEE Trans. Netw. Service Manag.*, vol. 18, no. 1, pp. 526–537, Mar. 2021.
- [21] D. Basu, A. Jain, U. Ghosh, and R. Datta, "QoS-aware dynamic controller implantation over vSDN-enabled UAV networks for real-time service delivery," in *Proc. 4th ACM MobiCom Workshop Drone Assist. Wireless Commun. 5G Beyond*, 2021, pp. 37–42.
- [22] B. Nour et al., "A survey of Internet of Things communication using ICN: A use case perspective," *Comput. Commun.*, vol. 142–143, pp. 95–123, 2019.
- [23] G. Xylomenos et al., "A survey of information-centric networking research," *IEEE Commun. Surv. Tut.*, vol. 16, no. 2, pp. 1024–1049, Apr.–Jun. 2014.
- [24] T. Kitagawa, S. Ata, and M. Murata, "Retrieving information with autonomously-flying routers in information-centric network," in *Proc. IEEE Int. Conf. Commun.*, 2016, pp. 1–6.
- [25] P. Boccadoro, M. Losciale, G. Piro, and L. A. Grieco, "A standard-compliant and information-centric communication platform for the internet of drones," in *Proc. Eur. Wireless 24th Eur. Wireless Conf.*, 2018, pp. 1–6.
- [26] Y. Gao, T. Kitagawa, S. Ata, S. Eum, and M. Murata, "On the use of naming scheme for controlling flying router in information centric networking," in *Proc. IEEE Int. Conf. Commun. Workshops*, 2018, pp. 1–6.
- [27] E. Barka, C. A. Kerrache, R. Hussain, N. Lagraa, A. Lakas, and S. H. Bouk, "A trusted lightweight communication strategy for flying named data networking," *Sensors*, vol. 18, no. 8, 2018, Art. no. 2683.
- [28] M. Kaur, R. Amin, and J. Martin, "The design and validation of ICN-Enabled hybrid unmanned aerial system," in *Proc. IEEE Mil. Commun. Conf.*, 2021, pp. 169–174.
- [29] Z. Ullah, F. Al-Turjman, U. Moatasim, L. Mostarda, and R. Gagliardi, "UAVs joint optimization problems and machine learning to improve the 5G and beyond communication," *Comput. Netw.*, vol. 182, 2020, Art. no. 107478.

- [30] V. Sharma, F. Song, I. You, and H.-C. Chao, "Efficient management and fast handovers in software defined wireless networks using UAVs," *IEEE Netw.*, vol. 31, no. 6, pp. 78–85, Nov./Dec. 2017.
- [31] G. Secinti, P. B. Darian, B. Canberk, and K. R. Chowdhury, "SDNs in the sky: Robust end-to-end connectivity for aerial vehicular networks," *IEEE Commun. Mag.*, vol. 56, no. 1, pp. 16–21, Jan. 2018.
- [32] M. A. Alharthi, A. M. Taha, and H. S. Hassanein, "An architecture for software defined drone networks," in *Proc. IEEE Int. Conf. Commun.*, 2019, pp. 1–5.
- [33] P. Berde et al., "ONOS: Towards an open, distributed SDN OS," in *Proc. 3rd Workshop Hot Topics Softw. Defined Netw.*, A. Akella and A. G. Greenberg, Eds. Chicago, Illinois, USA: ACM, 2014, pp. 1–6.
- [34] "The linux foundation. OpenDaylight project," [Online]. Available: <https://www.opendaylight.org/>
- [35] A. Hermosilla, A. M. Zarca, J. B. Bernabe, J. Ortiz, and A. Skarmeta, "Security orchestration and enforcement in NFV/SDN-aware UAV deployments," *IEEE Access*, vol. 8, pp. 131779–131795, 2020.
- [36] F. Xiong et al., "Energy-saving data aggregation for multi-UAV system," *IEEE Trans. Veh. Technol.*, vol. 69, no. 8, pp. 9002–9016, Aug. 2020.
- [37] B. Liu, W. Zhang, W. Chen, H. Huang, and S. Guo, "Online computation offloading and traffic routing for UAV swarms in edge-cloud computing," *IEEE Trans. Veh. Technol.*, vol. 69, no. 8, pp. 8777–8791, Aug. 2020.
- [38] N. McKeown et al., "Openflow: Enabling innovation in campus networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, 2008.
- [39] K. J. White, E. Denney, M. D. Knudson, A. K. Mamerides, and D. P. Pezaros, "A programmable sdn nfv-based architecture for UAV telemetry monitoring," in *Proc. 14th IEEE Annu. Consum. Commun. Netw. Conf.*, 2017, pp. 522–527.
- [40] C. Pan, J. Yi, C. Yin, J. Yu, and X. Li, "Joint 3D UAV placement and resource allocation in software-defined cellular networks with wireless backhaul," *IEEE Access*, vol. 7, pp. 104279–104293, 2019.
- [41] Y. Bi et al., "Software defined space-terrestrial integrated networks: Architecture, challenges, and solutions," *IEEE Netw.*, vol. 33, no. 1, pp. 22–28, Jan./Feb. 2019.
- [42] K. Chen, S. Zhao, N. Lv, W. Gao, X. Wang, and X. Zou, "Segment routing based traffic scheduling for the software-defined airborne backbone network," *IEEE Access*, vol. 7, pp. 106162–106178, 2019.
- [43] Z. Zhao et al., "Software-defined unmanned aerial vehicles networking for video dissemination services," *Ad Hoc Netw.*, vol. 83, pp. 68–77, 2019.
- [44] B. Li, Z. Fei, and Y. Zhang, "UAV communications for 5G and beyond: Recent advances and future trends," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2241–2263, Apr. 2019.
- [45] G. Secinti, A. Trotta, S. Mohanti, M. Di Felice, and K. R. Chowdhury, "FOCUS: Fog computing in UAS software-defined mesh networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 6, pp. 2664–2674, Jun. 2020.
- [46] K. Lei, Q. Zhang, J. Lou, B. Bai, and K. Xu, "Securing ICN-based UAV ad hoc networks with blockchain," *IEEE Commun. Mag.*, vol. 57, no. 6, pp. 26–32, 2019.
- [47] K. A. Raza, A. Asheralieva, M. M. Karim, K. Sharif, M. Gheisari, and S. Khan, "A novel forwarding and caching scheme for information-centric software-defined networks," in *Proc. Int. Symp. Netw., Comput. Commun.*, 2021, pp. 1–8.
- [48] Z. Latif et al., "DOLPHIN: Dynamically optimized and load balanced path for inter-domain SDN communication," *IEEE Trans. Netw. Serv. Manag.*, vol. 18, no. 1, pp. 331–346, Mar. 2021.
- [49] "NDN packet format specification 0.3," [Online]. Available: <https://named-data.net/doc/NDN-packet-spec/current/index.html>
- [50] A. Bujari, C. E. Palazzi, and D. Ronzani, "A comparison of stateless position-based packet routing algorithms for FANETS," *IEEE Trans. Mobile Comput.*, vol. 17, no. 11, pp. 2468–2482, Nov. 2018.
- [51] J. F. Shortle, J. M. Thompson, D. Gross, and C. M. Harris, *Fundamentals of Queueing Theory*, vol. 399. Hoboken, NJ, USA: Wiley, 2018.
- [52] J. D. Little and S. C. Graves, "Little's law," in *Building Intuition*. Berlin, Germany: Springer, 2008, pp. 81–100.
- [53] G. Baier, E. Köhler, and M. Skutella, "The k-splittable flow problem," *Algorithmica*, vol. 42, no. 3, pp. 231–248, 2005.
- [54] D. P. Bertsekas and D. A. Castanon, "A forward/reverse auction algorithm for asymmetric assignment problems," *Comput. Optim. Appl.*, vol. 1, no. 3, pp. 277–297, 1992.
- [55] L. Zhu et al., "SDN controllers: A comprehensive analysis and performance evaluation study," *ACM Comput. Surv.*, vol. 53, no. 6, pp. 133:1–133:40, 2021.
- [56] NSNAM, "NS-3, A discrete-event network simulator for internet systems," 2021. [Online]. Available: <https://www.nsnam.org/>

- [57] S. Mastorakis, A. Afanasyev, and L. Zhang, "On the evolution of ndnSIM: An open-source simulator for NDN experimentation," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 47, no. 3, pp. 19–33, 2017.



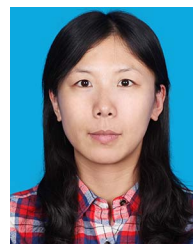
**Liehuang Zhu** (Senior Member, IEEE) received the Ph.D. degree in computer science from the Beijing Institute of Technology, Beijing, China, in 2004. He is currently a Professor with the School of Cyberspace Science and Technology, Beijing Institute of Technology. His research interests include security protocol analysis and design, group key exchange protocols, wireless sensor networks, cloud computing, and blockchain applications.



**Md Monjurul Karim** received the B.Eng. and M.Eng. degrees in computer science from Northwestern Polytechnical University, Xian, China. He is currently a Research Assistant with the Southern University of Science and Technology, Shenzhen, China. He is currently working toward the Ph.D. degree in computer science and technology with the Beijing Institute of Technology, Beijing, China. His research interests include software-defined networking, information-centric networking, multi-access edge computing, and next-generation networking.



**Kashif Sharif** (Senior Member, IEEE) received the M.S. degree in information technology in 2004, and the Ph.D. degree in computing and informatics from the University of North Carolina in Charlotte, NC, USA, in 2012. He is currently an Associate Research Professor with the Beijing Institute of Technology, Beijing, China. His research interests include information centric networks, artificial intelligence, blockchain and distributed ledger technologies, wireless and sensor networks, software defined networks, and data center networking. He is an Associate Editor for the several IEEE Journals.



**Chang Xu** (Member, IEEE) received the Ph.D. degree in computer science from Beihang University, Beijing, China, in 2013. She is currently an Associate Professor with the School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing. Her research interests include security and privacy in VANET, and Big Data security.



**Fan Li** (Member IEEE) received the B.Eng. and M.Eng. degrees in communications and information system from the Huazhong University of Science and Technology, Wuhan, China, the M.Eng. degree in electrical engineering from the University of Delaware, Newark, DE, USA, and the Ph.D. degree in computer science from the University of North Carolina in Charlotte, NC, USA. She is currently a Professor with the School of Computer Science, Beijing Institute of Technology, Beijing, China. Her research interests include wireless networks, smart sensing, crowd sensing, and mobile computing. Her papers was the recipient of the Best Paper awards from the IEEE MASS in 2013, the IEEE IPCCC in 2013, the ACM MobiHoc in 2014, and the Tsinghua Science and Technology in 2015.