February 2024
Master’s Degree Thesis

Joint Optimization of Data Aggregation and Offloading for UAV-Aided IoT Systems

Graduate School of Chosun University

Department of Computer Engineering

Asif Mahmud Raivi

Joint Optimization of Data Aggregation and Offloading for UAV-Aided IoT Systems

무인 비행체 활용 사물 인터넷 시스템을 위한 데이터 집계 및 오프로딩의 공동 최적화

February 23, 2024

Graduate School of Chosun University

Department of Computer Engineering

Asif Mahmud Raivi

Joint Optimization of Data Aggregation and Offloading for UAV-Aided IoT Systems

Advisor: Prof. Sangman Moh, Ph.D.

A thesis submitted in partial fulfillment of the requirements for a master's degree

October 2023

Graduate School of Chosun University

Department of Computer Engineering

Asif Mahmud Raivi

라이비 아시프 마무드의
석사학위논문을 인준함

위원장 신석주 (인) 申錫柱印

위 원 강문수 (인) 姜文秀印

위 원 모상만 (인) 牟相晩印

2023년 12월

조선대학교 대학원

Table of contents

List of Figures

List of Tables

Abstract

Joint Optimization of Data Aggregation and Offloading for UAV-Aided IoT Systems

Asif Mahmud Raivi
Advisor: Prof. Sangman Moh, Ph.D.
Department of Computer Engineering
Graduate School of Chosun University

In recent years, unmanned aerial vehicles (UAVs) have been used to extend the Internet of things (IoT) framework owing to their vast applications, monitoring and surveillance capability, ubiquity, and mobility. To support IoT requirements, UAVs must be capable of aggregating, processing, and transmitting data in real-time basis. As not only the number of IoT devices but also the amount of data to be collected has increased, data aggregation is of great importance. Recently, the UAV can also function as a mobile edge computing server in association with aerial data aggregation.

Owing to high flexibility and rapid deployment, unmanned aerial vehicles (UAVs) can provide network coverage for IoT devices in post-disaster scenarios. UAV-aided mobile edge computing (MEC) corroborates computational support as well as optimal decision-making process for ground IoT devices. Nonetheless, both data aggregation and computational offloading have been separately studied in the existing literature. In this thesis, therefore, we propose a joint data aggregation and computational offloading (JDACO) scheme in UAV-enabled IoT for post-disaster scenarios. JDACO focuses on minimizing the total cost of energy and delay in aggregation and computation process by utilizing UAV as MEC server and deploying multiple UAVs. Firstly, designing our objective function to evaluate the cost associated with aggregation and offloading process. Then, the optimization problem as Markov model is presented and multi-agent deep reinforcement learning algorithm is adopted, in which value decomposition with double deep Q-Network algorithm allows optimal data aggregation and cost-effective offloading process by utilizing the cooperative learning process. The experiment results reveal that the proposed JDACO scheme outperforms the existing schemes in terms of training time, computed data, energy consumption, and mission time while serving the maximum number of IoT devices.

요 약

무인 비행체 활용 사물 인터넷 시스템을 위한 데이터 집계 및 오프로딩의 공동 최적화

라이비 아시프 마무드
지도교수: 모상만
컴퓨터공학과
조선대학교 대학원

최근 무인 비행체(UAV)는 방대한 응용 분야, 모니터링 및 감시 기능, 편재성, 이동성 등으로 사물 인터넷 프레임워크를 확장하기 위해 사용되고 있다. 사물 인터넷(IoT) 요구사항을 지원하기 위해서는 UAV 가 실시간으로 데이터를 집계, 처리 및 전송할 수 있어야 하고, IoT 기기의 수뿐만 아니라 수집해야 할 데이터의 양이 증가함에 따라 데이터 집계의 중요성이 매우 커지고 있으며, 최근에는 항공 데이터 집계와 연계하여 모바일 엣지 컴퓨팅 서버로서의 기능도 수행하고 있다.

높은 유연성과 신속한 배치로 인해 무인 비행체는 재난 이후 상황에서 IoT 장치에 대한 네트워크 커버리지를 제공할 수 있다. UAV 지원 모바일 에지 컴퓨팅(MEC)은 지상 IoT 장치에 대한 최적의 의사 결정 프로세스뿐만 아니라 계산 지원을 제공한다. 그럼에도 불구하고, 데이터 집계와 계산 오프로딩은 각각 별도로 연구되어 오고 있다. 본 연구에서는 재난 이후 상황에 대한 UAV 활용 IoT 시스템에서 데이터 집계 및 계산 오프로딩의 공동 최적화(JDACO) 기법을 제안한다. JDACO 는 UAV 를 MEC 서버로 활용하고 여러 UAV 를 배치하여

데이터 집계 및 계산 과정의 총 에너지 비용과 지연을 최소화하는 데 중점을 둔다. 먼저, 데이터 집계 및 오프로딩 과정과 관련된 비용을 평가하는 목표 함수를 설계한다. 이어서 Markov 모델로서의 최적화 문제를 제시하고 이중 심층 Q-네트워크 알고리즘을 사용하여 최적의 데이터 집계 및 비용 효율적인 오프로딩 과정을 허용하는 다중 에이전트 심층 강화 학습 알고리즘을 채택한다. 실험 결과에 따르면, 제안한 JDACO 기법은 최대 수의 IoT 장치를 지원하면서 훈련 시간, 계산 데이터, 에너지 소비 및 임무 수행 시간 측면에서 기존의 방법들보다 우수하다.

1. Introduction

1.1 Overview

Recent technological advancements have led to unprecedented changes in the support roles played by unmanned aerial vehicles (UAVs) or drones, which were originally applied to military [1] and surveillance [2], [3] operations. Their low cost and ubiquity [4], coupled with their basic working principles [5] of fast deployment, high mobility [6], ease of use, line-of-sight (LoS) connectivity [7], [8] and reliable data communications [9], have led to their advanced support for civilian communications [10], [11]. Applications now include wildfire monitoring [10], consumer payload delivery [12]–[14] image recognition [15], aerial base-station (BS) relay [16], search and rescue [17], and smart agriculture [18]. By acting as bridges between cyber and physical layers, they have unlocked seemingly limitless possibilities for autonomous operations [19]. Hence, it is no surprise that they are now being considered for extending Internet of thing (IoT) networks [20], which will further facilitate our society's low data footprint and federated processing needs [21], [22].

Rapid advent in wireless communication networks and Internet of things (IoT) has made terrestrial communication possible [23]. Unmanned aerial vehicles (UAVs) have opened new avenues for communication technology [24]. UAVs will soon become an integral part of existing communication systems owing to their easy and rapid deployment. The use of UAVs is increasing from military missions to industrial and commercial applications [14], [25]. Recently, UAVs have been proposed for restoring communications in post-disaster scenarios [26]. Therefore, UAVs are expected to become powerful and important entities for shaping communication systems in the near future.

According to a report published by Statista.com [27], the number of connected IoT devices is projected to reach more than 29 billion by 2030, three times more than the number of connections in 2020. Moreover, with recent developments in communication technology in association with 5G and beyond, it has been made possible to integrate fast moving UAV-based services into the existing communication system [28]. This increase is naturally expected to generate far more traffic than current networks can handle. Augmented by extant BSs, UAVs can assume the roles of computing centers or edge servers to process sensed data from ground IoT devices while reducing delays [29]–[32].

The implementation of new technologies poses various challenges. UAVs have limited battery capacity, resulting in limited flight time and need to be replenished before the next deployment. Therefore, to ensure smooth operation during the mission, the UAV flight trajectory should be carefully designed. Additionally, IoT devices installed for environmental monitoring are resource-constrained with limited computational capabilities and are often installed in hard-to-reach areas with the expectation of a long service life. Thus, any disruption in the existing communication systems can defeat the entire purpose of installing IoT devices. Moreover, because of their limited energy, IoT devices cannot communicate over long distances. Therefore, a well-planned strategy is required to maintain a stable connectivity between IoT devices and base stations (BS).

Owing to the rapid deployment capability of UAVs with extended battery life resulting from recent technological advancements, they can perform aggregation missions and edge units to support data-driven IoT applications [33]. There are several approaches in the literature in which

UAV collect data from ground IoT devices [34]. These studies primarily focused on the optimal point of data gathering, trajectory design for the UAV, devising an energy-saving scheme, resource allocation, and reducing the data collection period while ensuring the quality of service (QoS), data freshness, maximum data collection, and reduced loss of aggregated data.

Similarly, considering UAV as edge servers, the existing literature focuses on minimizing the task execution delay and energy requirements while ensuring maximum throughput and computation capability within the available edge server resources [35]. If the computation requirement is beyond the processing power, all or some of the computations are offloaded to another server with a higher computation capability, such as the BS. Consequently, IoT nodes can eliminate the burden of computation and perform for long periods of time. UAVs are perfect suitors for solving communication and computational issues resulting from both natural and man-made disasters. Additionally, urban air mobility (UAM) is drawing a lot of attraction both in industry and academia equally [36],[37].

The use of UAV as data aggregators has attracted considerable attention in recent years. Existing studies have focused on finding the optimal hovering location or cluster head selection for aggregating data from ground IoT nodes, as designing optimal path planning is essential for UAVs while ensuring minimal travel time and energy efficiency [38]. Similarly, UAVs are considered as an edge unit for offloading the computation-incentive tasks of IoT nodes, in which either binary or partial offloading is exploited in existing works [39]. Thus, IoT nodes are protected from heavy computation and long service times. Existing studies primarily focus on latency and energy minimization for the offloading process, while ensuring maximum data computation and throughput maximization. Although many studies have considered static and single-UAV scenarios, recent studies have focused on UAV mobility and multi-UAV deployment [39].

1.2 Research Objective

In the existing literature, it can be observed that data aggregation and computational offloading were studied separately. Recognizing the future prospects of UAVs for data-driven applications, a joint data aggregation and computation offloading scheme is proposed and introduced the two problems under the same umbrella, rather than considering them as separate problems. A more detailed study of the existing literature has been discussed in the related works section.

To address the aforementioned discussions and limitations, a proposal of joint multi-UAV-based data aggregation and computation offloading scheme to mitigate the overall system cost of the process has been offered. More explicitly, multiple UAVs are deployed, where each UAV is responsible for data aggregation and computation, as well as offloading some computation to another UAV or BS with higher resources and computational capability. The introduction of a multiagent paradigm brings the action and decision-making of each UAV into unison, as all agents share their experiences with each other. The key contributions of this study are as follows:

- A study of joint scenario of data aggregation and computation offloading from a data-driven aerial computing perspective, which has not yet been explored for UAV-enabled services.
- A joint data aggregation and computational offloading (JDACO) scheme mathematically for a multi-UAV scenario is developed. The proposed optimization problem primarily focuses on minimizing the total cost of energy consumption and delay for the aggregation and offloading

processes, while ensuring maximum IoT device coverage.

- To address the joint optimization problem, a multi-agent reinforcement learning-based algorithm is proposed in which a dueling double deep Q network (D3QN) for the discrete action space is adopted and a decision maker for each UAV. Value decomposition network (VDN) algorithm is employed for cooperative learning among the UAVs.
- The proposed algorithm is evaluated using two other learning algorithms and one non-learning algorithm in terms of key performance metrics. Simulation results demonstrate the superiority of the proposed algorithm over other benchmarks.

1.3 Thesis Layout

The remainder of this thesis is organized as follows:

Firstly, the relevant studies that have been conducted thus far in the respective fields of data aggregation and task offloading are explored in Section 2. The proposed system model is presented in Section 3 and the optimization problem is formulated in Section 4. In Section 5, the formulated optimization problem is transformed into a Markov game model. In Section 6, the performance of the proposed JDACO algorithm is demonstrated and compared with that of other benchmarks. Finally, the conclusion of this thesis is discussed in Section 7.

2. Related Works

Most studies on UAV-aided data-driven applications can be categorized into two main classes. The first focuses on utilizing UAV as relays or base stations (BS) to provide a backbone for data-gathering applications. In such cases, the UAV is considered a data aggregator, where the trajectory of the UAVs is designed based on the communication schedule [40]. In the second class, UAVs act as mobile edge computing (MEC) units to support the computational capabilities of a given network [41].

2.1 Existing Data Aggregation Techniques in UAV-Aided IoT

UAV is performing a key integral role of aerial data aggregation [42]. A brief description of each of the roles of UAV-aided IoT data aggregation is discussed as follows.

A. Direct Aggregation

UAV can perform aggregation tasks by itself, using its onboard processing capability. A typical scenario would involve UAV loitering over a collection of designated ground nodes in the area of interest. Trajectories and waypoints can be applied based on the arrangement of nodes, their computational power, etc. After aggregation, the UAV sends the aggregated data to a ground device for further processing. In most cases, a single UAV is deployed at a time. There are several advantages to this scenario. For example, if the UAV can perform aggregation, the ground IoT devices will require less processing and fewer computations, which extends their lifetimes. However, UAVs must be loaded with greater storage and computational power, causing them to require more visits to charging stations.

B. Aggregation as Collector

UAVs can collect aggregated or raw sensing data from distributed ground IoT devices and CHs in the field of interest and relay them to sinks or BSs. Flight plan and trajectory considerations are roughly the same as before, as are the advantages for ground nodes, in that they are not burdened with additional processing requirements. The UAV, on the other hand, does not require extra computational and processing power as it is solely responsible for data collection and relay.

C. Aggregation as Sink

UAVs can be deployed as aerial sinks or BSs, in which they perform the roles of aggregation and extended processing. To fulfill this role, UAVs must have very strong onboard computational and processing power. This is very beneficial to regions of interest lacking communication infrastructures (e.g., post-disaster recovery). In such cases, the need for multiple simultaneous UAVs would be greater, further increasing the logistical complexity.

D. Hybrid Aggregation

UAVs will likely perform a combination of the roles described above, even simultaneously. Naturally, the computational power and processing capability would need to be very robust, as would the flight planning capacity of the command-and-control infrastructure. Fig. 3 presents a role-based aggregation scheme in which different UAV types are deployed in rural and urban areas to perform related aggregated tasks. In the following UAV-enabled data aggregation techniques are extensively reviewed. The various UAV roles in IoT data aggregation are exemplified across many literatures.

ML approaches to data aggregation and processing are quite popular among disparate research fields. However, for aerial data aggregation, there is currently a lack of sufficient data that can be used for model training. To address this issue, DRL is a viable option for handling unstructured or unknown features in which knowledge discovery is expected during operations. Unlike supervised learning, in which a labeled dataset is trained for model prediction, and unsupervised learning, in which unlabeled data are analyzed on the fly, DRL interacts with unknown datasets from scratch as they explore the region of interest, working toward a desired algorithmic reward by applying a trial-and-error method. For UAV flight planning, DRL can be used to design optimal trajectories and data-aggregation procedures.

Federated learning (FL) is another approach as it benefits from decentralized training data consumption, which allows the model to experientially learn from iterative activities. To apply FL to UAV data aggregation, edge processing units are often required, either aboard the UAV or via the ground relay. Notably, during operations, information security and communications protocols are vital planning and design considerations.

Parameter optimization techniques can be classified into two categories: deterministic and stochastic. Deterministic algorithms are based on a set of specific rules for solving problems, whereas stochastic ones follow probabilistic transition rules.

Inspired by the trajectory control and device communication protocol dilemmas faced by aerial data aggregators, a DRL-based DDPG scheme called ODDPG was proposed in [43]. In authentic environmental conditions, the states of IoT devices (e.g., battery level, queue backlog, and channel state) will likely be unknown to the UAV. The DDPG scheme prevents memory overflow and maximizes data aggregation as the UAV cruises among widespread ground IoT devices. In the example of [43], the researchers used Google's TensorFlow to take the joint UAV trajectory and data communication schedule as discrete-time partially observable Markov decision processes, and waypoints and battery levels were formulated as states. The desired action was then to locate the next waypoint and IoT device. The model reward was the aggregated data collected from IoT devices. Because the DDPG model exploits an actor–critic neural network algorithm, mini-batch gradient descent is applied as the loss with respect to UAV speed, packet arrival schedule, and IoT device channel condition for network training and instantaneous trajectory updating. *Lessons learned*: In-flight trajectory updating is quite challenging when real-time data are required owing to the inherent fallibility of DRL and FL schemes. However, DDPG provides better packet-loss results than if the free-walk model (the basic random baseline) were used. Moreover, classification errors can be overcome if the model is trained for longer periods. However, DDPG and policy-based gradient methods typically have slow convergence rates; hence, the model performs better when IoT node information is known in advance. Based on the application scenario, some of the superior machine learning approaches such as soft actor-critic (SAC) and proximal policy optimization (PPO) can be explored. Combining SAC with DDPG approach can maximize exploration by incorporating entropy into objective function leading to robust policies whereas PPO utilizes multiple updates of data for smooth and stable learning.

The IDAS scheme was formulated in [44] to restore post-disaster communications among IoT devices using DRL for UAV flight planning and associated tasks. The scheme relies on a tradeoff between data aggregation and energy costs. For training, a state–action reward is sampled from the UAV's previous flight record in mini-batches, and another actor–critic configuration is applied using an update factor to improve learning stability. The scheme used four deep neural networks to produce optimized aerial data aggregation schemes and achieved the highest data aggregation rates under variable UAV speeds among all benchmarks. However, increasing the speed leads to missed nodes and very high energy consumption on behalf of the UAV. Notably, this method is quite suitable for reestablishing communications and aerial data aggregation capabilities after a disaster. Rather than using simple system model to describe a post-disaster data aggregation scenario, a detailed model is necessary, which considers device activation and non-overlapping data aggregation for the effective and efficient use of UAV-enabled data aggregation service.

Blockchain-based secure schemes are reliable when dealing with sensitive data. An FL-based blockchain-embedded data accumulation scheme using drones for IoT (FBI) was proposed in [45], in which a two-step data authentication method was applied. The first validation mechanism used a cuckoo filter, and the subsequent mechanism used a Hampel filter to calculate the loss function. Trained models with local IoT data were used to prevent unauthorized access during training. A long short-term memory was used for task preparation, and the UAV maintained NLOS communication most of the time.

The study of arbitrary spatial patterns is known as stochastic geometry and is used to predict communication network performance. UAV-based IoT services are used to predict the coverage and energy consumption of multiple aggregators. Research in [46] demonstrated how the communications spectrum can be time-division multiplexed among ground user equipment (UE), terrestrial BSs, and UAVs when the locations of ground nodes are known. The available frequencies were validated using stochastic geometry. Notably, both UE and IoT device coverages were considered during UAV flight planning for energy management and flight altitude.

Ensuring the longest and most optimized operation times for IoT sensor nodes is challenging. For UAV data aggregation, the research in [47] applied a genetic algorithm (GA), which is one of the popular heuristic approaches, to find optimal solutions under both constrained and unconstrained conditions by imitating the natural selection processes of biological evolution. At each iteration (generation), the GA selects an individual from the population to become a parent of the next round of offspring. This evolutionary process continues until an optimal performing generation is reached. A greedy algorithm comprising two subproblems was applied to manipulate the spatial correlations between energy-efficient UAV flight planning and IoT device scheduling, based on their locations and activation schedules. Because the aggregation points were determined by adopting the nearest neighbor scheme, the TSP problem was solved.

ICS [48] offers a collision-free data aggregation scheme based on incremental clustering that coordinates UAV motions and trajectories. Notionally, the CH is offered sufficient time to collect data for aggregation prior to being visited by the UAV. Owing to this aggregation delay, the maximum number of nodes can be harvested. Owing to incremental clustering and scheduling scheme that calculates the transmission schedule of sensors based on the UAV's trajectory and velocity, allowing cluster heads visited later to have more time to collect data. This approach significantly reduces the data aggregation time.

It is often necessary to minimize the total flight and loitering times of UAVs, and the authors in [49] developed a mathematically mixed-integer NP-hard problem to arrive at a decoupled heuristic solution. For problem formulation, IoT devices were placed across a predefined area according to the Poisson point process. The area was then further divided into random subregions. In this scenario, UAVs communicated with IoTs using a slotted ALOHA random-access protocol in which activation messages were received by both parties. A multi-cooperative UAV trajectory was designed in which loitering locations were formulated using stochastic geometry. A decoupled heuristic approach was then used to address the NP-hard mixed-integer problem, replacing it with an approximation of the closed-form traveling period.

While considering aerial data aggregation scenarios, the existing studies have primarily focused on designing optimal trajectories by finding the optimal hovering point, minimizing mission energy, and covering the maximum number of IoT devices for aggregation. In [40], the authors studied mission cost minimization while covering the maximum number of IoT devices using multiple UAV as aggregators. A heuristic approach was used to solve the proposed problem. This study considers an IoT device activation model for an intruding probability scenario of communication between a UAV and IoT nodes.

In [50], a UAV was employed as a data aggregator, and the aggregated data were relayed to a base station (BS). In addition to the impact of the UAV altitude on the aggregation rate, the data-to-overhead ratio was studied to measure the effectiveness of data aggregation using UAV. The study mentions the possibility of further utilization of UAV as an edge unit to minimize end-to-end delay but does not explore that option.

The study in [49] aims to minimize the hovering and travelling time for data aggregation by a UAV visiting each node using a decoupled heuristic approach. This study demonstrates that a clear trade-off between hovering and travelling times is necessary for optimal data aggregation.

In [51], the struggle between the trajectory and data aggregation based on device activation in a multi-UAV scenario was studied. The concept of shared observation among UAV uses a long short-term memory (LSTM) deep deterministic policy gradient (DDPG) approach. The scheme addresses the pressing issues of data loss owing to buffer overflow and communication failures that may occur at ground IoT nodes.

2.2 Existing Computation Offloading Techniques in UAV-MEC

Computation of the sensing data is necessary because IoT nodes are reconstructed and have very limited computation capability. Edge units or devices are often introduced to address the computation problem and reduce the overall latency and energy consumption of the offloading process [52].

The study in [39] study UAV as MEC edge units, where the offloading process was classified into two categories: binary and partial offloading. The main idea behind utilizing UAV as edge units is to reduce the overall delay and energy consumption for IoT devices with resource-intensive tasks.

Moreover, in hierarchical aerial networks such as the space-air-ground integrated network (SAGIN), every component of each layer is considered an MEC unit and is able to perform resource-intensive tasks.

In [53], a partial offloading mechanism was proposed for a hierarchical network, where the IoT and UAV game-theoretic-based offloading decision method was suggested. Additionally, a heuristic approach was proposed to make offloading decisions between the UAV and a high-altitude platform (HAP). To utilize all the layers to the fullest extent possible, an adjustment algorithm is introduced. However, the mobility of the UAV and HAP is static, and the detailed mechanism of task collection has not been studied.

In [54], another partial offloading mechanism was studied, in which UAV mobility was considered. The proposed method aims to minimize the overall delay and energy requirements of the offloading process, while maximizing the number of arrival tasks. This study considers the local processing and task queuing delays in the total processing delay calculation. The algorithmic approach utilizes multi-objective reinforcement learning (MORL) to obtain the optimal solution. However, their proposed problem was demonstrated using only a single UAV.

In [55], a binary offloading problem was proposed, which was solved by a multi-agent actor-critic approach that aims to solve multiple objectives such as offloading decisions, flight direction, and distance. The proposed model has a relatively simple sensing model that is unrealistic considering the real environment. Considering all the matters at hand, we propose the JDACO scheme for maximum IoT device coverage with minimal energy and time expenditure for both the data aggregation and computation offloading processes.

2.3 Comparison of Existing Aggregation and Offloading Algorithms in UAV-MEC

In this section, we discuss a comparative summary of the existing works which shows a clear distinction between this work. As mentioned earlier, many of the existing works on UAV aided data-driven applications consider either data aggregation or computation aggregation as a separate entity or as part of the assumption. In this work, both UAV-aided data aggregation and computation offloading are studied equally and joint problem is formulated to address the issue. **Table 1.** demonstrates the key distinction between existing works vs this work.

Table 1. Comparative summary of existing works and ours

Ref.	Major focus		Algorithm		Number of UAVs	Optimization objective	Considered factors			
	Data aggregation	Offloading	DRL	Non-DRL			Trajectory and hovering location	Energy	Delay	Max number of IoT devices
[40]	✓	–	–	✓	Multiple	Max device coverage with min travel time	✓	✓	–	✓
[49]	✓	–	–	✓	Multiple	Min aggregation delay	✓	–	✓	✓
[50]	✓	–	–	✓	Single	Min aggregation delay	–	–	✓	–
[51]	✓	–	✓	–	Multiple	Min travel time with device scheduling	✓	–	✓	–

Ref.	**Major focus**		**Algorithm**		**Number of UAVs**	**Optimization objective**	**Considered factors**			
	Data aggregation	**Offloading**	**DRL**	**Non-DRL**			**Trajectory and hovering location**	**Energy**	**Delay**	**Max number of IoT devices**
[52]	–	✓	✓	–	Multiple	Energy and delay minimization	–	✓	✓	–
[53]	–	✓	–	✓	Multiple	Max data computation	–	✓	✓	✓
[54]	–	✓	–	✓	Single	Min energy and delay and max task computation	–	✓	✓	✓
[55]	–	✓	✓	–	Multiple	Min energy and delay for task computation	✓	✓	✓	–
Ours	✓	✓	✓	–	Multiple	Max device coverage, task with min energy and delay	✓	✓	✓	✓

Note: ✓ Studied; – Not studied

3. System Model and Problem Formulation

In this section, the system model of a multi-UAV-aided MEC for UAV-enabled IoT is presented. After introducing the application scenario, mobility, communication, data aggregation, local computation, and offloading computation models were formally addressed.

3.1. Motivation Scenario

In this thesis, a post-disaster region where multiple UAVs are deployed to aggregate data from live homogeneous IoT nodes on the ground is considered, as existing communication infrastructure such as base stations (BS) is no longer available. The deployed IoT nodes are responsible for monitoring environmental conditions, and low-tier UAVs (LT-UAV) are responsible for aggregating and offloading data based on the task size. Because the existing communication network has been disrupted, a UAV with a longer flight time and computation power, called a high-tier UAV (HT-UAV), hovers at a fixed altitude (which is higher than the altitudes of LT-UAVs) such that all the LT-UAVs are under the communication coverage of the HT-UAV.

Figure 1 shows a typical example of the network configuration in the application scenario. To aggregate data from ground IoT devices, LT-UAVs must hover over several hovering locations where data can be collected from the maximum number of IoT devices. As LT-UAVs aggregate data from IoT devices, each LT-UAV flies for the maximum travel time of $T_{\max}$ before the maximum energy $E_{\max}$ of the UAV is depleted, and then lands on the ground.

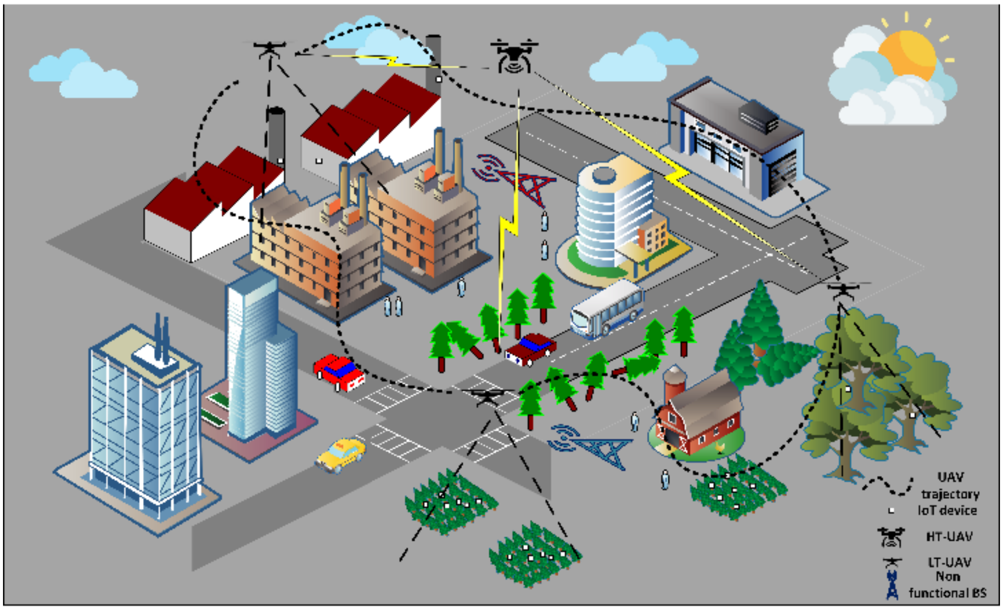


Figure 1. Application scenario of typical network configuration.

An assumption is that, during the hovering mode, each LT-UAV flies at a pre-defined average velocity of V_{avg} and $V = 0$ m/s. The ground node locations are known beforehand and are distributed statically over the area of interest. Node locations can be expressed as $i = [x_i, y_i]$. To avoid collisions with other UAVs or foreign objects, each UAV has object detection capabilities, thereby ensuring a safe flight plan. IoT devices are static and randomly distributed across geographical areas. Each UAV maintains a considerable altitude to ensure strong line-of-sight (LoS) communication in urban regions. Additionally, the HT-UAV ensures the synchronized trajectory of other LT-UAVs for both non-overlapping aggregation locations and the estimation of the number of active IoT devices in the area of interest. **Table 2** lists the key notation and definitions of the respective notation used in the entire thesis.

Table 2. List of key notations with respective definition

Notation	Definition
$P_{\text{LoS}}^{i,j}, P_{\text{NLoS}}^{i,j}$	Line-of-sight (LoS) and non-line-of-sight (NLoS) probability between IoT node *i* and LT-UAV *j*.
$L_{\text{LoS}}^{i,j}, L_{\text{NLoS}}^{i,j}$	Path loss for LoS and NLoS condition
$\Lambda_{i,j}$	Average path loss between IoT node *i* and LT-UAV *j*.
$\mathbb{S}_{i,j}$	Signal-to-interference-plus-noise-ratio (SINR) between IoT node *i* and LT-UAV *j*.
$R_{i,j}$	Expected data rate between IoT node *i* and LT-UAV *j*.
$G_{j,k}$	Channel gain between LT-UAV *j* and HT-UAV *k*.
$\mathbb{S}_{j,k}$	Signal-to-interference-plus-noise-ratio (SINR) between LT-UAV *j* and HT-UAV *k*.
$R_{j,k}$	Uplink data rate between LT-UAV *j* and HT-UAV *k*.
N_i	The number of active IoT nodes ready for transmitting data.
$\mathring{A}_{i,j}, \tilde{A}_{i,j}$	Indicator function for establishing communication and multiple transmissions between IoT node *i* and LT-UAV *j*.
$\Phi_{j,k}$	Binary decision variable for local computation and offloading between LT-UAV *j* and HT-UAV *k*.

3.2. LT-UAV Mobility Model

To ensure a strong LoS and avoid obstacles in urban regions we assume that the LT-UAVs fly at a considerable altitude of h_j. The horizontal direction and distance travelled by the LT-UAV at time slot t is denoted as $\emptyset(\mathrm{t})$ and $d(\mathrm{t})$, respectively, provided the following conditions are satisfied:

$$0 \leq \emptyset(\mathrm{t}) \leq 2\pi, \;\; 0 \leq d(\mathrm{t}) \leq d_{\max}, \tag{1}$$

where d_{max} is the maximum flying distance of the LT-UAV owing to its limited battery capacity.

We adopted a conventional Cartesian coordinate system to represent the mobility of the UAV. Let $\mathcal{U}(\mathrm{t}) = \left[x_{\mathrm{j}}(\mathrm{t}), y_{\mathrm{j}}(\mathrm{t})\right]$ represent the LT-UAV's location at time slot t. Thus, based on the $\emptyset(\mathrm{t})$ and $d(\mathrm{t})$, the coordinate of the LT-UAV at the next time slot $\mathrm{t} + 1$ can be expressed as

$$\begin{cases} x_j(t+1) = \ x_j(t) + d(t) \cdot \cos(\emptyset(t)) \\ y_j(t+1) = y_j(t) + d(t) \cdot \sin(\emptyset(t)) \end{cases}. \tag{2}$$

The LT-UAV was assumed to travel within an enclosed rectangular region with side lengths are $x_{\max}$ and $y_{\max}$. We have

$$0 \leq x_j(t) \leq x_{\max}\,, \ \ 0 \leq y_j(t) \leq y_{\max}. \tag{3}$$

Similar to previous studies [54], [56], we adopted the propulsion power requirement of a rotary-wing UAV to define its power consumption, which is given by

$$P(v(t)) = P_1\left(1 + \frac{3v(t)^2}{U_{\mathrm{tip}}^2}\right) + P_2\left(\sqrt{1 + \frac{v(t)^4}{4v_0^4}} - \frac{v(t)^2}{2v_0^2}\right)^{1/2}$$
$$+\frac{1}{2}d_0\rho g A v(t)^3 \tag{4}$$

The given equation comprises three components: blade profile, induced power, and parasitic power. P_1 is the blade profile power in the hovering state, and P_2 is the induced power. U_{tip} refers to the speed of the rotor blade tip and v_0 is the average induced rotor velocity during the hovering state. The power of the parasite was also contained. d_0, ρ, g, and A which are fuselage drag ratio, density of air, solidity of the rotor, and disk area, respectively. Under hovering conditions, the power consumption of the UAV is an aggregation of P_1 and P_2. The overall energy requirement of the UAV during its flight duration T is given by

$$E_j^{\mathrm{fly}}(t) = \sum_0^T P\big(v(t)\big)\Delta t. \tag{5}$$

3.3. Communication Model

The communication model is formulated into two different segments: communication between the IoT node and LT-UAV and communication between the LT-UAV and HT-UAV.

A. Downlink Communication Model

As stated previously, the LT-UAV maintains a considerable altitude to maintain a strong LoS. Therefore, the LoS probability between the ground IoT node i and LT-UAV j can be expressed as

$$P_{\mathrm{LoS}}^{i,j} = \frac{1}{1+\alpha e^{-\beta(\theta_{i,j}-\alpha)}}, \tag{6}$$

where α and β are the environmental constant values and the elevation angle, respectively, and $\theta_{i,j} = (\frac{180}{\pi})\sin^{-1}(\frac{h_j}{d_{i,j}})$, where $d_{i,j}$ denotes the distance between IoT node i and LT-UAV j and can be expressed as $d_{i,j} = \sqrt{((x_j(t) - x_i)^2 + (y_j(t) - y_i)^2 + h_j^2))}$. As expected, non-line-of-sight (NLoS) probability is $P_{\mathrm{NLoS}}^{i,j} = 1 - P_{\mathrm{LoS}}^{i,j}$. The path-loss expression for both LoS and NLoS is expressed as

$$L_{\mathrm{LoS}}^{i,j} = \eta_{\mathrm{LoS}}(\frac{4\pi f_{\mathrm{c}}}{c} d_{i,j})^{\xi} \tag{7}$$

and

$$L_{\mathrm{NLoS}}^{i,j} = \eta_{\mathrm{NLoS}}(\frac{4\pi f_{\mathrm{c}}}{c} d_{i,j})^{\xi}, \tag{8}$$

respectively, where η_{LoS} and η_{NLoS} are the attenuation factor for LoS and NLoS state, respectively, f_c is the carrier frequency, c is the speed of light, and ξ is the path loss component. Thus, the average path loss $\Lambda_{i,j}$ between IoT node i and LT-UAV j can be found as

$$\Lambda_{i,j} = P_{\mathrm{LoS}}^{i,j} \times L_{\mathrm{LoS}}^{i,j} + P_{\mathrm{NLoS}}^{i,j} \times L_{\mathrm{NLoS}}^{i,j}. \tag{9}$$

Therefore, average channel gain at time instant t is $G_{i,j}(t) = {\Lambda_{i,j}}^{-1}$. It is assumed that each IoT node i has a transmit power $P_i(t)$ at time instant t and the IoT devices communicate with the LT-UAV via a time-division multiple access (TDMA) scheme. Employing the TDMA scheme eliminates intra-cluster interference. However, neighboring UAV may cause interference. Considering these factors, the signal-to-interference-plus-noise-ratio (SINR) between IoT node i and LT-UAV j at time instant t is expressed as

$$\mathbb{S}_{i,j}(t) = \frac{P_i(t)\, G_{i,j}(t)}{P_n(t) G_{m,n}(t) + \sigma^2}\ n{=}1, m{\neq}i, n{\neq}j, \tag{10}$$

where σ^2 denotes the Gaussian noise variance. Using Shannon's theorem, we calculated the approximate data rate between IoT node i and LT-UAV j, which is denoted as

$$R_{i,j}(t) = \mathcal{B}_1\, log_2(1 + \mathbb{S}_{i,j}(t)) \tag{11}$$

where $\mathcal{B}_1$ is the channel bandwidth for the downlink communication.

B. Uplink Communication Model

Because the existing communication infrastructure, such as the BS, is no longer available in the post-disaster scenario, LT-UAVs are resource-constrained and need to offload the aggregated data to the HT-UAV, which has higher processing power and computation capacity. Assuming that the wireless link between the LT-UAV and HT-UAV maintains clear LoS characteristics, the channel quality depends on the instantaneous distance between them [57]. Let $\mathcal{v} = [x_{\mathrm{k}}, y_{\mathrm{k}}]$ denote HT-UAV coordinates. Then, the instantaneous distance between LT-UAV j and HT-UAV k is given as $d_{\mathrm{j,k}} = \sqrt{||\mathcal{v} - \mathcal{U}(\mathrm{t})||^2}$. Therefore, the channel power gain between the LT and HT-UAV, following the path loss model in free space at time instant $t,$ can be expressed as

$$G_{j,k}(t) = \mathcal{P}_0 d_{j,k}^{-2} = \frac{\mathcal{P}_0}{||\mathcal{v} - \mathcal{U}(t)||^2}, \tag{12}$$

where $\mathcal{P}_0$ is the power gain of the channel at 1 m distance and is subjected to the antenna gain and carrier frequency.

Because we intend to maintain communication between LT-UAVs and HT-UAV continuously, we exploit the benefit of the frequency division multiple access (FDMA) scheme. The uplink bandwidth $\mathcal{B}_2$ is divided into J non-overlapping sub-bands of J LT-UAVs. Thus, in each time slot, each LTUAV uplink was allotted a subband of $\frac{\mathcal{B}_2}{J}$. Then, the SINR can be formulated as

$$\mathbb{S}_{j,k}(t) = \frac{P_j(t) G_{j,k}(t)}{\frac{\mathcal{B}_2}{J}\sigma_0^2} = \frac{P_j(t)\mathcal{P}_0}{\frac{\mathcal{B}_2}{J}||\mathcal{v} - \mathcal{U}(t)||^2 \sigma_0^2}, \tag{13}$$

where $P_j(\mathrm{t})$ is the transmission power of the LT-UAV and σ_0^2 is the spectrum power density of white Gaussian noise (WGN) at the HT-UAV. Similar to Equation (10), we can calculate the uplink data rate using Shannon's theorem:

$$R_{j,k}(t) = \frac{\mathcal{B}_2}{J} log_2(1 + \mathbb{S}_{j,k}(t)). \tag{14}$$

3.4. Data Aggregation Model

For data aggregation from ground IoT nodes using a UAV, a definitive IoT device activation pattern is essential for designing appropriate waypoints and optimal hovering location [49].

A. IoT Device Activation Model

Monitoring IoT sensors such as smart metering are usually accompanied by periodic activation, whereas event-driven IoT sensors such as wildfire monitoring follow random activation scenarios [40]. The central server has prior information regarding the periodic activation conditions. Thus, the periodic IoT device activation model for the number of active IoT devices, $\mathcal{N}_{\text{act}}$, over period $[0, T]$ can be expressed as

$$\mathcal{N}_{\text{act}} = \frac{T}{\tau_i}, \tag{15}$$

where τ_i is the period during which the IoT device is active. In the case of randomly activated IoT devices that are often subjected to bursty traffic and short activation intervals, we incorporate the probability density function of the random activation model, $\mathcal{N}_{\text{act}}$, as studied in [40] over the time period $[0, T]$, which is defined as

$$D(t) = \frac{t^{a-1}(T-t)^{\mathscr{b}-1}}{T^{a+\mathscr{b}-1}\, B(a,\mathscr{b})}, \tag{16}$$

where $B(a,\mathscr{b}) = \int_0^1 t^{a-1}\,(1-t)^{\mathscr{b}-1} dt$ is the beta function with parameters denoted as a and $\mathscr{b}$, and are known as shape parameters provided $a, \mathscr{b} \geq 0$. Using both periodic and random activation models is a crucial design consideration because we aim to determine the optimal hovering location for maximal data aggregation.

B. Aggregation Cost Calculation

The selection of an appropriate aggregation location is a prerequisite for energy-saving. In our work, we aim to find the optimal hovering location, where the maximum number of IoT devices can be served based on the received SINR at LT-UAV j. The number of active IoT devices at any given time can be expressed as $|N_i| = N_i^{\mathrm{per}} + N_i^{\mathrm{rand}}$. A more elaboration is

$$N_i(t) = \sum_{i=1}^{W_i^{\mathrm{per}}} \varphi_i^{\mathrm{per}}(t) + \sum_{i=1}^{W_i^{\mathrm{rand}}} \varphi_i^{\mathrm{rand}}(t), \tag{17}$$

where $\varphi_i^{\mathrm{per}}(t)$ and $\varphi_i^{\mathrm{rand}}(t)$ are binary functions and defined as

$$\varphi_i^{\mathrm{per}}(t) = \begin{cases} 1, & \text{if } i \text{ is active at } t_{\mathrm{p}} \\ 0, & \text{otherwise} \end{cases}. \tag{18a}$$

and

$$\varphi_i^{\mathrm{rand}}(t) = \begin{cases} 1, & \text{if } D(t) \geq D(t_{\mathrm{th}}) \\ 0, & \text{otherwise} \end{cases}. \tag{18b}$$

respectively.

Furthermore, if the SINR value reaches certain threshold, $\mathbb{S}_{\mathrm{i,j}}^{\mathrm{th}}$, then the IoT node establishes communication with LT-UAV and $\mathbb{S}_{i,j}(t) \geq \mathbb{S}_{i,j}^{\mathrm{th}}$. Therefore, the indicator function can be defined as:

$$\mathring{A}_{i,j}(t) = \begin{cases} 1, \text{if } \mathbb{S}_{i,j}(t) \geq \mathbb{S}_{i,j}^{\mathrm{th}} \\ 0, \quad \text{otherwise} \end{cases}. \tag{19}$$

To avoid multiple communications and ensure that one IoT node communicates with a particular LT-UAV simultaneously, we introduce another indicator function:

$$\tilde{A}_{i,j}(t) = \begin{cases} 1, \text{if } \mathring{A}_{i,j}(t) = 1 \text{ and } \varphi_i = 1 \\ 0, \qquad\qquad \text{otherwise} \end{cases}. \tag{20}$$

Therefore, the modified expression for the data rate in Equation (10) is transformed into:

$$R_{i,j}(t) = \mathring{A}_{i,j}(t)\tilde{A}_{i,j}(t)\mathcal{B}_1 \, log_2(1 + \mathbb{S}_{i,j}(t)). \tag{21}$$

Finally, the time period for aggregating data at a particular hovering point of the LT-UAV can be expressed as

$$T_j^{\text{agg}} = \sum_{i=1}^{W_i^{\text{per}}} \varphi_i^{\text{per}} (\frac{S_i}{R_{i,j}(t)}) + \sum_{i=1}^{W_i^{\text{rand}}} \varphi_i^{\text{rand}} \left(\frac{S_i}{R_{i,j}(t)}\right). \tag{22}$$

where S_i is the data or task size collected from the IoT devices, and the energy during hovering can be expressed as

$$E_j^{\text{agg}}(t) = \sum_0^{T_j^{\text{agg}}} P\big(v(t)\big)\Delta t. \tag{23}$$

3.5. Local Computation Model

After all data from the IoT nodes are aggregated by the LT-UAV, the onboard processing unit starts processing the data locally. Similar to the local processing model in [55], the internal unit of each LT-UAV was equipped with a single-core CPU. Thus, the LT-UAV can execute only one task at a time, and the remainder is offloaded to the HT-UAV for further processing. Thus, the queuing delay was not considered in the proposed model. Because IoT nodes are homogenous, the task size is uniform, and the number of arriving tasks is the same as the number of active IoT nodes at a particular time, $N_i(t)$. The duration of the local computing can be expressed as

$$T_j^{\mathrm{loc}}(\mathrm{t}) = \sum_{j=1}^{\mathrm{J}}(1 - \Phi_{j,k}\,)\frac{S_i V_j}{f_j^{\mathrm{loc}}}, \tag{24}$$

where f_j^{loc} is the local CPU frequency of the LT-UAV, and V_{j} is the required CPU cycle to complete the task. $\Phi_{j,k} \in [0, 1]$ is the binary decision variable for either local execution or offloading to the HT-UAV, and can be represented as

$$\Phi_{j,k} = \begin{cases} 1, & \text{if offloaded} \\ 0, & \text{locally computed} \end{cases}. \tag{25}$$

We can compute the energy expanded for the local computation as

$$E_j^{\mathrm{loc}}(t) = P_j^{\mathrm{loc}} \times T_j^{\mathrm{loc}}(t) = P_j^{\mathrm{loc}} \sum_{\mathrm{j}=1}^{\mathrm{J}}(1 - \Phi_{\mathrm{j,k}})\frac{S_i V_{\mathrm{j}}}{f_{\mathrm{j}}^{\mathrm{loc}}}, \tag{26}$$

where P_j^{loc} is the power requirement for local computation and is proportional to the cubic power of the local frequency f_j^{loc} of the LT-UAV [58]. The resulting equation is as follows:

$$E_j^{\mathrm{loc}}(t) = \mu(f_{\mathrm{j}}^{\mathrm{loc}})^2 \sum_{\mathrm{j}=1}^{\mathrm{J}}(1 - \Phi_{\mathrm{j,k}}) S_i V_{\mathrm{j}}, \tag{27}$$

where μ is the LT-UAV’s effective capacitance factor subjected to the CPU chip architecture.

3.6. Offloading Computation Model

Considering the limited resources and computational capacity of LT-UAV, tasks are offloaded to HT-UAV, which are equipped with higher processing power and edge units. Therefore, the transmission time from an LT-UAV j to an HT-UAV k can be written as

$$T_j^{\mathrm{tran}}(t) = \sum_{\mathrm{j}=1}^{\mathrm{J}} \sum_{n=0}^{N_i-1} (\Phi_{j,k}) \frac{S_i V_k}{R_{i,j}(t)}. \tag{28}$$

Similarly, the energy requirement for the data transmission is expressed as

$$E_j^{\mathrm{tran}}(t) = P_j^{\mathrm{tran}} \times T_j^{\mathrm{tran}}(t), \tag{29}$$

where P_j^{tran} is the transmission energy required for offloading. Similar to the previously defined duration of the local computation, we define the offloading computation time as

$$T_j^{\mathrm{Off}}(\mathrm{t}) = \sum_{j=1}^{\mathrm{J}} \sum_{n=0}^{N\mathrm{i}-1} (\Phi_{j,k}) \frac{S_i V_k}{f_k^{\mathrm{off}}}, \tag{30}$$

where f_k^{off} denotes the CPU frequency of the HT-UAV. Similarly, the energy expanded for offloading can be calculated as follows:

$$E_k^{\mathrm{off}}(t) = P_j^{\mathrm{off}} \times T_j^{\mathrm{Off}} = P_j^{\mathrm{off}} \sum_{j=1}^{\mathrm{J}} \sum_{n=0}^{N\mathrm{i}-1} (\Phi_{\mathrm{j,k}}) \frac{S_i V_k}{f_k^{\mathrm{off}}}, \tag{31}$$

where P_j^{off} denotes the processing power required for the HT-UAV. As mentioned previously, the resulting equation becomes

$$E_j^{\mathrm{loc}}(t) = \mu(f_{\mathrm{k}}^{\mathrm{off}})^2 \sum_{\mathrm{j}=1}^{\mathrm{J}} \sum_{n=0}^{N\mathrm{i}-1} (\Phi_{\mathrm{j,k}}) S_i V_j. \tag{32}$$

3.7. Energy and Delay Cost Calculation

According to all the defined equations, we can obtain the total cost of energy and delay associated with the proposed problem. Therefore, the overall energy cost associated with the joint data aggregation and offloading processes can be expressed as:

$$E_{\mathrm{tot}}(t) = E_j^{\mathrm{fly}}(t) + E_j^{\mathrm{agg}}(t) + E_j^{\mathrm{loc}}(t) + E_j^{\mathrm{tran}}(t) + E_k^{\mathrm{off}}(t). \tag{33}$$

Similarly, the total delay can be expressed as

$$T_{\mathrm{tot}}(t) = T_j^{\mathrm{agg}}(t) + T_j^{\mathrm{loc}}(t) + T_j^{\mathrm{tran}}(t) + T_k^{\mathrm{off}}(t) \tag{34}$$

3.8. Problem Formulation

Our objective is to design an optimized algorithm for the overall data aggregation and offloading processes while serving the maximum number of IoT nodes. Based on an earlier discussion, our goal is to minimize both the total energy consumption of LT-UAVs and the total aggregation and task execution time. Task execution time is the elapsed time for local execution and offloading. The total aggregation and task execution time is the time elapsed from the beginning of the data aggregation (i.e., the first transmission of the data to be aggregated from the IoT devices) to the end of the computation offloading (i.e., the last reception of the task to be offloaded). To define the optimization problem, we normalized E_{tot} and T_{tot} as $E_n = E_{\mathrm{tot}}/E_{\max}$ and $T_n = T_{\mathrm{tot}}/T_{\max}$, respectively, where $E_{\max}$ and $T_{\max}$ are the maximum values of E_{tot} and T_{tot}, respectively. The optimization problem can be formulated as follows:

$$\mathrm{P}_1: \quad \min_{N_{\mathrm{i}}, H} \omega_1 E_{\mathrm{n}} + \omega_2 T_{\mathrm{n}} \tag{35}$$

$$s.t. E_{\mathrm{tot}} \leq E_{\mathrm{th}}, \tag{C1}$$

$$0 \leq \emptyset(\mathrm{t}) \leq 2\pi, \tag{C2}$$

$$0 \leq d(\mathrm{t}) \leq d_{\max}, \tag{C3}$$

$$\mathbb{S}_{\mathrm{i,j}}(\mathrm{t}) \geq \mathbb{S}_{\mathrm{i,j}}^{\mathrm{th}}, \tag{C4}$$

$$0 \leq \mathrm{\mathring{A}}_{\mathrm{i,j}}(\mathrm{t}) \leq 1, \tag{C5}$$

$$0 \leq \tilde{A}_{\mathrm{i,j}}(\mathrm{t}) \leq 1, \tag{C6}$$

$$S_{\mathrm{i}} \leq S_{\mathrm{i(max)}} \tag{C7}$$

$$0 \leq \Phi_{\mathrm{j,k}} \leq 1, \tag{C8}$$

where ω_1 and ω_2 are the weight parameters for the total energy requirement of LT-UAVs and the total aggregation and task execution time, respectively, and $\omega_1 + \omega_2 = 1$. Based on the mission requirement, the parameters ω_1 and ω_2 can be adjusted. Constraint C1 ensures that each UAV does not exceed the maximum threshold energy available for the duration of the mission. Constraints C2 and C3 are UAV movement constraints. Constraint C4 ensures the optimal hovering location selection based on the received SINR between the UAV and IoT nodes. where C5 is the indicator constraint for C4, Constraint C6 ensures that each IoT node can be connected simultaneously to a particular LT-UAV. Constraint C7 ensures that the UAV buffer memory is not overflowed by incoming data. C8 is the offloading constraint between LT-UAVs and HT-UAV, which depends on the computational capability of LT-UAVs.

To select the optimal hovering location H for data aggregation, we introduced several constraints on the UAV mobility. Therefore, we introduce another optimization problem for solving the UAV trajectory problem, which can be expressed as

$$\mathrm{P}_2: \quad \min_{H} \sum_{\mathrm{u}=1}^{U} \sum_{\mathrm{e}=0}^{E} \sum_{\substack{\mathrm{f}=0 \\ \mathrm{f}\neq\mathrm{e}}}^{F} \| h_{\mathrm{e}} - h_{\mathrm{f}} \| \, x_{\mathrm{ef}}^{\mathrm{U}} \tag{36}$$

$$s.t. \;\; \sum_{\mathrm{u}=1}^{\mathrm{U}} \sum_{\mathrm{e}=0}^{\mathrm{E}} x_{\mathrm{ef}}^{\mathrm{U}} = 1 \; \forall_{f} = 1, \dots, F, \; \mathrm{f} \neq \mathrm{i}, \tag{C9}$$

$$\sum_{\mathrm{e}=1}^{\mathrm{E}} x_{\mathrm{eg}}^{\mathrm{U}} - \sum_{\mathrm{f}=1}^{\mathrm{E}} x_{\mathrm{gf}}^{\mathrm{U}} = 0 \; \forall_{\mathrm{g}} = 1, \dots, G, \tag{C10}$$

$$\sum_{\mathrm{f}=1}^{\mathrm{F}} x_{0\mathrm{f}}^{\mathrm{U}} = 1 \; \forall_{\mathrm{f}} = 1, \dots, \mathrm{F}, \tag{C11}$$

$$\sum_{\mathrm{f}}^{\mathrm{F}} x_{\mathrm{f}0}^{\mathrm{U}} = 1 \; \forall_{\mathrm{f}} = \mathrm{F}, \tag{C12}$$

$$\vartheta_{h_{\mathrm{e}}} - \vartheta_{h_{\mathrm{f}}} + E \sum_{\mathrm{u}=1}^{\mathrm{U}} x_{\mathrm{ef}}^{\mathrm{U}} \leq E - 1, \tag{C13}$$

$$2 \leq \mathrm{e} \neq \mathrm{f} \leq E, \qquad \text{(C14)}$$

where, $x_{\mathrm{ef}}^{\mathrm{U}} \in \{0,1\}$ is a binary variable indicating LT-UAV’s movement for points e to f. Constraint C9 ensures that each hovering location is visited by UAV at least once. C10 ensures that each LT UAV leaves the same hovering point after aggregation. C11 and C12 indicate that the LT-UAV started its mission from the designated initial position and returned to the initial point after the completion of the mission. C13 and C14 are known as Miller-Tucker-Zemlin constraints [40], [59], which eliminate the subtour of LT-UAVs.

4. Joint Data Aggregation and Computation Offloading (JDACO)

In this section, a multi-agent deep reinforcement learning based (MA-DRL) approach is introduced to address proposed optimization problem P1 for joint data aggregation and computation offloading. We first model our original optimization problem as a Markov game, and then use the VD3QN approach [60] to solve the problem.

4.1. Markov Game Formulation

Because we deployed multiple UAV for joint data aggregation and offloading, each UAV's action was affected by the collaborative action of the other UAVs. Therefore, the proposed optimization problem can be transformed into a Markov game framework. The Markov game is an extension of the Markov decision process (MDP) for multi-agent scenarios [61]. A Markov game with N number agents can be designated as tuple (S, A, R, P), where S, A, R, P denote the state, action, reward function, and state-transition probability, respectively. At each time step, the agent η takes an action $a_\eta \in A$ based on a certain policy after observing the current environment state $s_\eta \in S$. A next state s' is chosen according to the state transition probability $P\left(s'_\eta \middle| s_\eta, a_1, a_2, a_3, \dots, a_\eta\right)$. By selecting the next state in an environment, reward r is obtained based on the reward function R. In terms of machine learning, a reward is simply a quantitative value that demonstrates the amount of an agent's action that has an impact on the agent's learning or objective.

We next discuss each component's definition for the Markov game formulation.

- Agent: Each LT-UAV is considered an agent as it begins interacting with the given environment and other LT-UAVs to maximize collaborative non-overlapping rewards and exchange information with each other. Therefore, the environment becomes fully observable, and each observation can be considered a state. Because each LT-UAV (agent) is deployed from the depot for data aggregation and makes local computations or offloading decisions, each UAV performs an appropriate action based on its respective policy. As each action is performed, a reward is generated from the environment and forwarded to the subsequent state. When each agent reaches its optimal goal, it stops receiving an additional reward from the environment or moves to the next state.
- State S: Because we deployed multiple LT-UAV in our simulation environment, our optimization problem can be described as a multi-agent Markov game. Every agent has its own state and acts independently of others. For the agent η, the state space, s_{η}, can be defined as

$$s_{\eta} = \{O_{\eta}, O_{-\eta}\}. \tag{37}$$

The state space has two components, the first component O_{η} is the self-observations of LT-UAV, whereas the second component $O_{-\eta}$ is the observation of the other LT-UAVs. The self-observations, O_{η}, can be defined as $O_{\eta} = \{b_{\eta}, E_{\eta}, \imath, \mathcal{U}_{\eta}, \mathbb{S}_{\eta}^{\mathrm{i,j}}, \tilde{A}_{\mathrm{i,j(\eta)}}, N_{\mathrm{i}}\}$ where b_{η} is the network identifier of each LT-UAV as we utilize the network-sharing method [62] and represented by one-hot vector, E_{η} is the remaining energy of the LT-

UAV, $\mathcal{i}$ is the location information of IoT nodes, $\mathcal{U}_{\mathrm{\eta}}$ is the location information, $\mathbb{S}_{\mathrm{\eta}}^{\mathrm{i,j}}$ is the SINR between i and j, $\tilde{A}_{i,j(\mathrm{\eta})}$ is the one-hot vector indexing indicating the UAV and IoT association and N_{i} is the task aggregated by LT-UAV to process. Similarly, $O_{-\mathrm{\eta}}$ is the shared observation resulting from the other agents in the environment and can be described as $O_{-\mathrm{\eta}} = \{\mathcal{i}, \mathcal{U}_{-\mathrm{\eta}}, \tilde{A}_{i,j(-\mathrm{\eta})}\}$ where $\mathcal{U}_{-\mathrm{\eta}}$is the location information of other LT-UAVs and $\tilde{A}_{i,j(-\mathrm{\eta})}$ is the device association parameter.

- Action A: Each agent requires an appropriate action in every time slot based on the current self and shared observations. The combined action of the agents can be expressed as $a_{\mathrm{\eta}} = \{a_M, \mathrm{Å}_{\mathrm{i,j(\eta)}}, \tilde{A}_{\mathrm{i,j(\eta)}}, x_{\mathrm{ef}}^{\mathrm{U}}, \Phi_{\mathrm{j(\eta),k}}\}$ where all the actions are taken in discrete action place and $\mathrm{Å}_{\mathrm{i,j(\eta)}}, \tilde{A}_{\mathrm{i,j(\eta)}}, x_{\mathrm{ef}}^{\mathrm{U}}, \Phi_{\mathrm{j(\eta),k}} \in \aleph$, $\aleph$ being the number of possible actions of $\mathrm{Å}_{\mathrm{i,j(\eta)}}, \tilde{A}_{\mathrm{i,j(\eta)}}, x_{\mathrm{ef}}^{\mathrm{U}}, \Phi_{\mathrm{j(\eta),k}}$ and are all binary variables. By integrating the LT-UAV mobility in horizontal direction, $\emptyset$ in discrete action space, the total number of possible actions for the LT-UAV is $2^{\aleph} \times \emptyset$.

- State Transition Probability P: The state of each agent or LT-UAV depends on its present location. Using equation (2) we can define the deterministic environment for the LT-UAV's position where the state transition probability for the next state of the agent is $P\left(s_{\mathrm{\eta}}' \middle| s_{\mathrm{\eta}}, a_1, a_2, a_3, \dots, a_{\mathrm{\eta}}\right) = 1$.

- Reward function R : As discussed earlier, the reward is the quantitative value received by an agent after interacting with the given environment, which numerically demonstrates how well the optimization objective has been achieved. The reward for the discrete time step can be defined as:

$$r_{\mathrm{t}} = r_{\mathrm{c}} + r_{\mathrm{e}} + r_{\mathrm{p}}\ . \tag{38}$$

As indicated in (34), the reward equation comprises the following three parts: The first part r_{c} is awarded to successfully complete the overall mission and is a positive number. The second part r_{e} is a violation constraint owing to the energy of the agent and the negative number. The final term is the penalty term r_p, which is also a negative number. The penalty term is $r_{\mathrm{p}} = r_{\mathrm{SINR}} + r_{\mathrm{ass}} + r_{\mathrm{mov}} + r_{\mathrm{off}}$, where r_{SINR} is the SINR constraint violation term, r_{ass} is the device association violation term, r_{mov} is the movement constraint violation term, and r_{off} is the offloading constraint violation term.

For each episode of time step τ, minimizing the overall energy and delay for the aggregation and offloading process, our proposed problem (P1) turns into maximizing the cumulative reward $G = \sum_{t=1}^{T} \sum_{\eta=1}^{N} r_t^{\eta}$. Therefore, the proposed Markov game formulation was an episodic task [63]. In every episode, the agent begins its journey from the initial state and ends in the terminal state by returning to its initial deployment position.

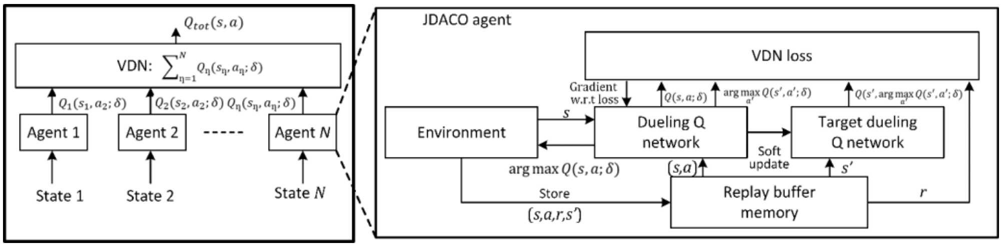


Figure 2. JDACO architecture and workflow

4.2. VD3QN Based Solution Approach

To solve our modified formulated problem, we adopted a learning-based algorithm called VD3QN [60], which is a combination of VDN [62] and a Dueling Double Deep Q Network (D3QN) [64] as shown in **Figure 2**. We modified the existing VD3QN algorithm to our advantage and modeled the proposed problem accordingly. The D3QN act as decision maker for each agent using the local action values $Q(s_\eta, a_\eta)$, whereas the VDN generates the global action value $Q_{\text{tot}}(s, a)$. Therefore, sequential optimal actions were achieved by achieving a common objective for an individual agent or LT-UAV. In the following, the D3QN and VDN architectures are studied to solve the formulated Markov game.

1) *D3QN***:** To obtain an optimal policy for an action-value function, the D3QN can be utilized as a value-based reinforcement learning technique [64]. Unlike standard DQNs [65], D3QN approximates the state value and state-dependent information first for each action taken and then perform aggregation function of the layers to obtain estimated action value function Q.

Each agent acquires observation of the environmental state s and utilizing the parameterized deep neural network (DNN) to produce action-value function,

$Q(s, a; \delta)$ which is an approximal value of the original action-value function, $Q(s, a)$. Moreover, utilizing the dueling architecture, the D3QN can quickly identify the Q value, which ultimately helps in a faster training process by choosing the appropriate action.

To learn the parameters of the neural network (NN), storing the state transitions (s, a, r, s') in experience replay buffer $\boldsymbol{B}$ plays a significant role, where s' is the next state after action a is performed and reward r is received in return. As the training phase continued, a minibatch of state transitions was randomly chosen from the replay memory buffer. Then, the parameters are brought up to date each time by reducing the square of the temporal difference (TD) error, which is given by

$$L(\delta) = \boldsymbol{E}_{s,a,r,s'}[\left(\boldsymbol{y}^{D3QN} - Q(s, a; \delta)\right)^{\mathbf{2}}]. \tag{39}$$

To address the overestimation problem of original Q-learning, we utilized the double Q-learning architecture [66], which is given by

$$\boldsymbol{y}^{\mathrm{D3QN}} = r + \varsigma Q\ (\mathrm{s}', \arg\min_{\mathrm{a}'} Q(\mathrm{s}', \mathrm{a}'\,; \delta); \delta^{\mathrm{t}}). \tag{40}$$

where ς is the discount factor and δ^{t} is the target parameters of the target neural network. It is to be noted that the architecture of the target network is same as action-value NN which obtains value from δ to ensure stable learning [67].

2) VDN Architecture**:** In the proposed model, each agent works for the common objective of maximizing the number of devices served while minimizing the overall energy and time. Value decomposition divides the value function of a multi-agent problem into separate value functions for each agent. This allows agents to learn to cooperate with each other because

they do not compete for the same resources. Therefore, all agents work independently and share their current state and observations cooperatively to find the global solution. Thus, we adopted the VDN [62] approach to find the global action-value function, which is denoted by Q_{tot}.VDN calculates the joint action-value function using the value-decomposition layer. Then, the summation of the action-value functions is calculated from the other agents, which is defined as

$$Q_{tot}(s, a) = \sum_{\eta=1}^{N} Q_{\eta}(s_{\eta}, a_{\eta}; \delta), \tag{41}$$

where s_{η} and a_{η} are each agent's state and actions respectively. By utilizing the value-decomposition layer, each agent can learn a better joint action in a noncompeting cooperative manner.

Algorithm 1 describes the proposed JDACO algorithm. In the training mode, every episode is defined by events where all agents start from the initial position, carry out aggregation, local computing, and offloading procedures, and then return to the initial position based on the remaining battery level. For each agent, each episode begins with $\tau = 0$ with initial state and reset all other parameters as defined in line 3. In line 4, we impose the maximum number of allowable steps to prevent an agent wandering around, and an energy condition is imposed to ensure that the agent does not fall off while wandering around. This is necessary as at the early stage of the training phase, agents have very little knowledge with a high probability of exploration, $\boldsymbol{\epsilon}$. Therefore, a new episode is initiated if an agent meets the desired target, or if a selected number of steps is reached. As the training phase continues, each agent encounters local state s_{η} (line 6). From line 7 to 8, based on the observed state, a random action is chosen from the action space using $\boldsymbol{\epsilon}$-greedy policy or using action-value function $Q(a_{\eta}, s_{\eta}; \boldsymbol{\delta})$ and then agent's location, energy information and other binary parameters are updated. As stated in line 10, the environment then generates reward r_{η} based on the prescribed reward formulation. As agents work in a cooperative manner, the sum of all the agent's reward is calculated as $r_{\text{total}} = \sum_{\eta} r_{\eta}$ (line 10). At this stage, the combined action a, current state s, following state s', and total reward r_{total} are recorded in the replay memory buffer $\boldsymbol{B}$. After that, time step τ and exploration rate $\boldsymbol{\epsilon}$ are updated as stated in line 13. Using the stored transition at the end, the DNN of the agent is trained, as stated in lines 15–18. More importantly, the $\boldsymbol{\delta}$ parameter is updated as loss function is minimized and denoted as:

$$L(\delta) = \frac{1}{\sqcap_{\mathbf{b}}} \sum_{\sqcap_{\mathbf{b}}} [(y_{\text{tot}} - Q_{\text{tot}}(s, a; \delta))^2], \tag{42}$$

with

$$y_{\text{tot}} = r_{\text{total}} + \varsigma Q_{\text{tot}}\,(\mathrm{s}', \arg\min_{\mathrm{a}'} Q_{\text{tot}}(s', a'\,; \delta); \delta^{\mathrm{t}}, \tag{43}$$

where $\sqcap_{\mathbf{b}}$ is the sampled episode number from replay buffer and $\boldsymbol{\delta}^{\mathbf{t}}$ are the parameters of the target NN. To stabilize the training process, $\boldsymbol{\delta}^{\mathbf{t}}$ are soft updated after every W episodes, as mentioned in line 18. Notably the aggregation operation is performed to sum the respective Q values and is not included in the parameters of the NN.

Algorithm 1: JDACO for energy and delay minimization

Input: Maximum episode number $\sqcap_{\mathbf{eps}}$, maximum step number per episode $\sqcap_{\mathbf{stp}}$, exploration at start $\boldsymbol{\epsilon}_{\mathbf{0}}$, decay rate $\boldsymbol{\varepsilon}$, achieving objective reward $\boldsymbol{r}_{\mathbf{o}}$, defilement reward $\boldsymbol{r}_{\mathbf{d}}$, replay buffer $\boldsymbol{B}$, batch size $\sqcap_{\mathbf{b}}$, Number of agent η, initial LT-UAV position $\boldsymbol{\mathcal{U}}_{\mathbf{0}}$, LT-UAV maximum energy $\boldsymbol{E}_{\max}$ IoT node location $\boldsymbol{i}$, Number of tasks $\boldsymbol{N}_{\mathbf{i}}$, initial parameters for NN $\boldsymbol{\delta}$, target parameters for NN $\boldsymbol{\delta}^{\mathbf{t}}$, rate of learning $\boldsymbol{\alpha}$, rate of soft update $\boldsymbol{\beta}$.

Output: Trained parameter $\boldsymbol{\delta}$.

```
Initialize $\boldsymbol{\delta}, \boldsymbol{\epsilon} = \boldsymbol{\varepsilon}_{\mathbf{0}}, \boldsymbol{\delta}^{\mathbf{t}} = \boldsymbol{\delta}$;
for $n_{\text{eps}} = 1,2,3,\dots,\sqcap_{\text{eps}}$ do
  Set timestep $\tau = 0$ and reset agent's position and other parameters
  $\mathcal{U}_{\eta}(\text{t}) = \mathcal{U}_{\eta}^{\text{init}}$ for each LT-UAV η;
  while $\mathcal{U}_{\eta}(\text{t}) \neq \mathcal{U}_{\eta}^{\text{fin}}$ and $E_{\eta} \leq \boldsymbol{E}_{\max}$ and $\tau \leq \sqcap_{\text{stp}}$ do
    for agent $\eta = 1,2,3,\dots,N$ do
      Get state $s_{\eta}$, based on agent's position;
      Selection of action $a_{\eta}$ from the defined action-space $A$
        using $\boldsymbol{\epsilon}$-greedy exploration policy, as
        $a_{\eta} = \begin{cases} \text{random action,} & \text{probabilistic } \boldsymbol{\epsilon} \\ \arg\max\limits_{a_{\eta} \in A} Q(a_{\eta}, s_{\eta}; \boldsymbol{\delta}), & \text{otherwise;} \end{cases}$
        Perform action $a_{\eta}$, update agent's position $\mathcal{U}_{\eta}(\text{t})$, agent's energy and other
      binary parameters
    end
    Calculate reward $r_t$ using equation (38) and obtain the cumulative reward $G$.
    Collect combined action $a$, current state $s$ and following state $s'$;
    Record the state transition information $(s, a, s', r_{\text{total}})$ in replay buffer $\boldsymbol{B}$;
    Update new timestep $\tau \rightarrow \tau + 1$ and new exploration rate $\boldsymbol{\epsilon} \rightarrow \boldsymbol{\varepsilon} \times \boldsymbol{\epsilon}$
  end
  Sample $\sqcap_{\mathbf{b}}$ episodes of minibatch from replay buffer $\boldsymbol{B}$;
  Obtain the loss function using equation (37)
  Using gradient descent optimizer, update $\boldsymbol{\delta}$
  $\boldsymbol{\delta} \rightarrow \boldsymbol{\delta} - \alpha \nabla_{\boldsymbol{\delta}} \big( y_{\text{tot}} - Q_{\text{tot}}(s, a; \boldsymbol{\delta}) \big)$;
  After every $W$episodes, update target parameter $\boldsymbol{\delta}^{\mathbf{t}}$ using soft update mechanism
  using,
  $\boldsymbol{\delta}^{\mathbf{t}} = (1 - \boldsymbol{\beta}) \boldsymbol{\delta}^{\mathbf{t}} + \boldsymbol{\beta} \boldsymbol{\delta}$;
end
```

4.3. Complexity Analysis

To analyze the complexity of the proposed scheme, we studied the time and space complexities of the DDQN training and VDN aggregation separately, and then studied the time and space complexities of our proposed JDACO algorithm based on the modified VD3QN. The time complexity of the DDQN architecture for experience collection is expressed by $O(\sqcap \times \sqcap_{\mathbf{stp}})$, where $\sqcap$ is the number of episodes during training phase and $\sqcap_{\mathbf{stp}}$ is the timesteps per episode. To update the replay buffer during training phase, the time complexity is denoted as $O(\sqcap_{\mathbf{b}} \times M)$, where $\sqcap_{\mathbf{b}}$ is the batch size and M is the number of iterations per episode. For the VDN aggregation, the time complexity can be given as $O(\eta \times A)$, where η being the number of agents and$A$ being the cardinality of the individual agent's action space. Thus, the time complexity of JDACO can be given as $O(\sqcap_{\mathbf{stp}} \times S \times A^{\mathbf{2}})$, where S being the number of states and A **is** the number of possible actions from the action space. This is because JDACO must explore all possible combinations of decision variables for all agents at each time step.

The space complexity of the DDQN is $O(B + P)$, where B and P are the replay buffer size and number of parameters in the network used for storing experiences and neural parameters, respectively. As for the VDN space complexity, it can be defined as $O(A \times \eta)$, The space complexity of JDACO can be expressed as $O(S \times A)$ as JDACO needs to store the Q-table, for every state and action.

5. Performance Evaluation

In this section, the performance of the proposed JDACO is evaluated by simulation results and compared to conventional schemes using TensorFlow framework version 1.15 on a desktop computer equipped with two 1070Ti processors with a total of 16GB of memory. To demonstrate the effectiveness of the proposed algorithm, we selected two learning-based approaches as our benchmarks: Q-learning with mixing (Qmix) [68] and counterfactual multi-agent policy gradients (COMA) [69]. We selected the heuristic greedy approach (HGA) as the nonlearning-based approach.

Similar to the VDN approach, Qmix is a value-based approach that utilizes centralized training decentralize execution method. It addresses the challenge of coordinating multiple agents to achieve a common goal by combining individual agent policies into a joint action-value function. We replaced the gated recurrent unit (GRU) with a D3QN unit to adopt Qmix in our problem. On the other hand, COMA is a deep reinforcement learning algorithm that uses counterfactual reasoning to assign credits to individual agents in cooperative multi-agent systems. It combines a centralized critic to evaluate the joint action value with individual actor networks that select actions for each agent based on local observations. To incorporate COMA into our formulated MDP, we replaced the GRU with multi-layer perceptron (MPL) NN. For the non-learning-based approach, we formulate the HGA as a binary integer programming problem by utilizing only the binary variables that are solved by the Python library called PuLP [70].

5.1. Simulation Setup

We performed the primary simulation by deploying multiple LT-UAVs from the initial coordinates (0, 0). The IoT nodes were randomly deployed over an area of (10 × 10 Km). The coordinates of the HT-UAV were (5, 5). As we aim to minimize both the energy and delay for the aggregation and computational offloading processes, we emphasize equal weights for both terms. The simulation parameters are listed in **Table 3**.

Table 3. Simulation Parameter

Parameter	Value
Simulation area	$10{\times}10\text{Km}^2$
η_{LoS} and η_{NLoS} values	1.6 dBm and 23 dBm
Carrier frequency,	2 GHz
Environment constant α and β	10.39, 0.05
Weight parameters ω_1 and ω_2	0.5 ($\omega_1 = \omega_2$)
UAV altitude, h_j	100 m
Number of IoT nodes	[20, 40, 60 80, 100 (default),]
Number of UAVs	[3, 5(default), 7]
Blade profile power, P_1	79.8563 W
Induced power, P_2	88.6279 W
Rotor blade tip speed, U_{tip}	120 m/s
Average induced velocity of rotor during hovering state, v_0	4.03 m/s
Fuselage drag ratio, d_0	0.6
Density of air, ρ	1.225 kg/m^3
Solidity of the rotor, g	0.05
Disk area, A	0.503 m^2
Effective capacitance factor, μ	10^{-28}
Task size	[2.5, 5 (default), 7.5, 10] Mbit
Channel power, $\mathcal{P}_0$	$1.42{\times}10^{-4}$
Local CPU frequency, $f_{\text{j}}^{\text{loc}}$	10^9 cycles/s
CPU cycle to finish the task, V_{j}	270 cycles/bit
CPU frequency of the HT-UAV, $f_{\text{k}}^{\text{off}}$	$5{\times}10^{10}$ cycles/s
Maximum episodes, $\sqcap_{\mathbf{eps}}$	1200
Maximum steps, $\sqcap_{\mathbf{stp}}$	60
Exploration probability at beginning	1
Mission completion reward, r_{c}	130
Energy violation reward, r_{e}	-20

Parameter	**Value**
Batch size, $\sqcap_{\mathbf{b}}$	64
Soft update, β	0.01
Learning rate, $\boldsymbol{\alpha}$	0.001
Discount factor, ℸ	0.99
Buffer size $\boldsymbol{B}$	500000

In the proposed JDACO architecture, we utilized a feed-forward, fully connected neural network with three hidden layers containing 256, 512, and 128 neurons. The neuron in the final layer corresponds to all possible actions that the agents can take. For simplicity, we consider three degrees of freedom (forward, left, and right) for each LT-UAV. The simulation values for the propulsion-power calculation of the LT-UAV were adopted from [71]. It is important to note that different factors, such as the number of agents and violation constraints, can have an impact on algorithm convergence.

In our simulation, the following performance metrics were evaluated: A brief description of each metric is provided below.

- **Average reward:** The performance indication of an agent over time helps to visualize the agent's learning interactions from the environment. It comprises the cumulative reward over time by improving the decision-making policy. An average reward curve or learning curve illustrates the fluctuating rewards as an agent explores different strategies to achieve convergence or stability of the learning process.
- **Total number of IoT nodes in service:** This performance metric indicates the active IoT nodes among all deployed IoT nodes that have successfully transmitted data to the LT-UAV for computation. Usually, a higher number suggests that the proposed scheme can achieve large

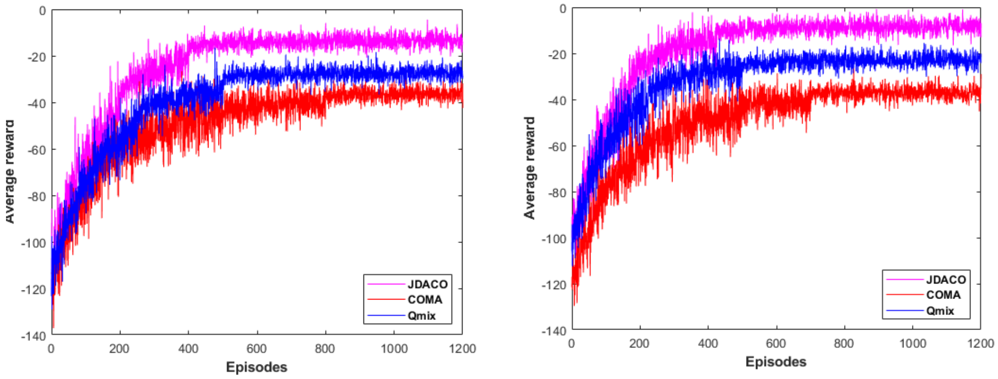
amounts of data without missing any IoT nodes that are ready to upload the data.

- **Total amount of computed data:** This indicates the total amount of data that an LT-UAV can gather for processing. A higher amount of data computation indicates that the LT-UAV was able to aggregate data from IoT nodes for computation or offloading without missing any of the nodes, which might result in data loss owing to the overflow of the buffer memory of IoT nodes.
- **Mission time:** This indicates the average time required for LT-UAVs to complete their journey, starting from deployment from the initial position, data aggregation time, data computation time, offloading time, and finally returning to the initial deployment position. The shorter time required for the predefined energy of the LT-UAV demonstrates the effectiveness of the proposed scheme.
- **Total energy consumption:** The total energy required for the LT-UAV to complete its mission, which includes travelling to the optimal hovering location, data aggregation, computation, and energy consumption offloading. Overall, lower energy consumption is an indication of an energy-efficient scheme.
- **Total aggregation and task execution time:** This refers to the time required to process data starting from the aggregation point when the UAV is hovering. In this state, the LT-UAV aggregates data from the ground nodes until no other nodes are ready to transmit the data. The execution time refers to the combination of local computation by the LT-UAV, transmission from the LT-UAV to the HT-UAV, and offloading by the HT-UAV. Because the delay for each data point is variable, we calculate the average delay for the overall aggregation, offloading, and

computation processes. In our proposed scheme, we ignore the queuing delay because the LT-UAV processes only a single task at a time, and the HT-UAV has sufficient processing power, making the queuing delay insignificant. The aggregation time for the LT-UAV is higher than the task execution time, as the LT-UAV collects and aggregates data from IoT nodes and is usually expressed in seconds. On the other hand, task execution usually takes a shorter time frame of approximately a few milliseconds.

5.2. Simulation Results and Discussion

First, the performances of the training processes of the DRL-based approaches is compared, as illustrated in **Figure 3**. The simulation results for the training process involved two instances with three LT-UAVs and five HT-UAVs for all DRL approaches. The results demonstrate the convergence of the algorithms for all instances. However, among learning-based approaches, COMA performed the worst. COMA operates under the principle of a counterfactual baseline mechanism, which inhibits the exploratory ability of the centralized critic. This renders COMA unsuitable for the proposed JDACO scheme. However, JDACO and Qmix show similarities in performance because both provide value-factorization-based solutions. The performance of the Qmix network can be improved by combining it with a more complex NN architecture and a global state with an action value. This is still unlikely to outshine the performance of JDACO because the local state of an agent has a full observation of all other agents, and further improvement is not guaranteed. Overall, JDACO reached convergence at a faster rate than the other baseline algorithms.

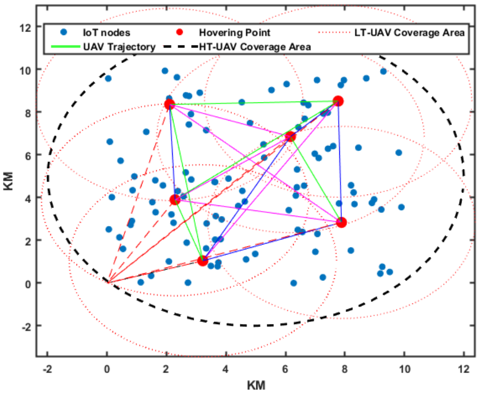


(a) **(b)**

Figure. 3. Average reward (a) with 3 LT-UAVs and (b) with 5 LT-UAVs.

Figure 4 shows the simulation scenario of our scheme with respect to an example deployment of HT-UAV, LT-UAVs, and IoT nodes. The IoT node distribution, LT-UAV coverage, HT-UAV coverage, and respective trajectories of the LT-UAVs are graphically shown. A simulated environment was generated using three LT-UAVs for 100 IoT nodes. The coordinates (0, 0) indicate the deployment points of the LT-UAVs, and coordinates (5, 5) indicate the positions of the HT-UAVs. Note that the negative distance is a vector representation of the simulation area.

In **Figure 5**, the performance of all benchmarks for IoT devices is explored. Compared to all the other benchmarks, HGA exhibited the poorest performance. This is understandable because a greedy approach aims to find the shortest way to finish the mission without prioritizing the number of IoT nodes in service and fulfilling the other constraints. On the other hand, among the learning-based benchmarks, JDACO and Qmix showed similar performances, whereas JDACO was superior to Qmix and COMA.



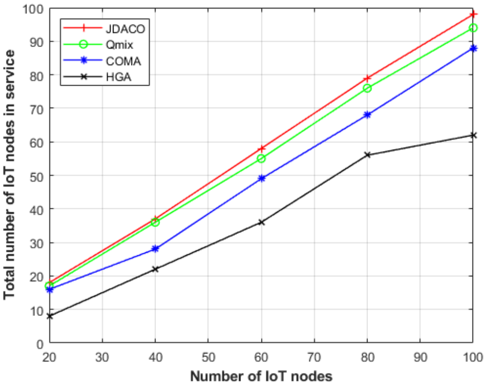


Figure 4. An example deployment of HT-UAV, LT-UAV and IoT nodes.

Figure 5. Total number of IoT nodes in service.

A further comparison of the amount of computed data among the different baseline algorithms is illustrated, as shown in **Figure 6**. For simplicity, the deployed nodes are sensory in nature is assumed, and each aggregated datum per sensor contains approximately 5 Mbits of data. It can be observed that JDACO computes more data than the other baseline approaches. This indicates that less data loss is ensured by the proposed JDACO scheme, whereas the other schemes fail to compute a portion of their data. This result also indicates the linear scalability of the proposed scheme compared to other baseline algorithms.

The mission times for the different schemes are compared while varying the number of LT-UAVs deployed, as illustrated in **Figure 7.** It is not surprising that the HGA scheme requires the longest time to complete the aggregation and offloading mission, as it must satisfy all conditions for aggregation and computation constraints. While it is true that increasing the number of UAVs reduces the overall mission time for all baseline schemes, JDACO requires the shortest mission time for all three use cases (i.e., for different numbers of

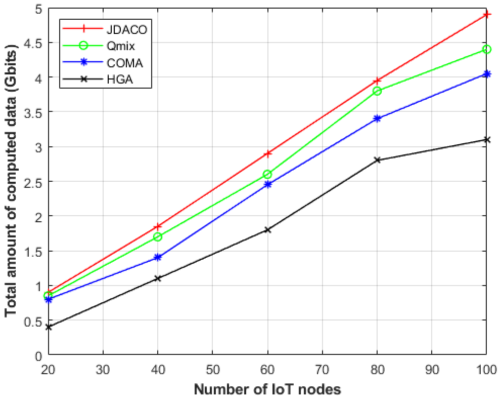


Figure 6. The total amount of computed data.

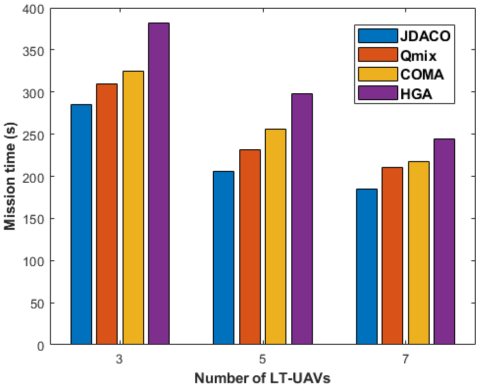


Figure 7. Mission time for different number of LT-UAVs.

LT-UAVs) among all other benchmarks.

The energy consumed by the LT-UAVs for different baseline schemes is studied. The energy expenditures for different CPU cycles for an LT-UAV with 100 IoT nodes is calculated. As shown in **Figure 8**, the proposed JDACO algorithm consumes less energy than the other learning-based algorithms. We also observed the impact of computational capability on the energy requirements. By proposing an energy-saving scheme, the UAVs can perform missions for a longer time in JDACO, which extends the scalability of the proposed scheme.

The aggregation and execution times for different task sizes are also evaluated and compared them with those of other benchmarks. The average aggregation and offloading times for each benchmark is explored because each LT-UAV has its own respective time based on observations from the environment. As seen from **Figure. 9**. Overall, the proposed JDACO scheme had less aggregation and execution time when matched with other benchmarks. The HGA has a higher time requirement for both instances.

Although Qmix demonstrates a performance similar to that of the proposed scheme, COMA requires longer aggregation and task execution times in both cases.

The impact of task size on performance is evaluated as well. In other words, the total energy consumption and mission execution time were observed by varying the task size. First, the energy consumed by the UAVs for different task sizes is studied. **Figure. 10** shows the energy consumed by the different

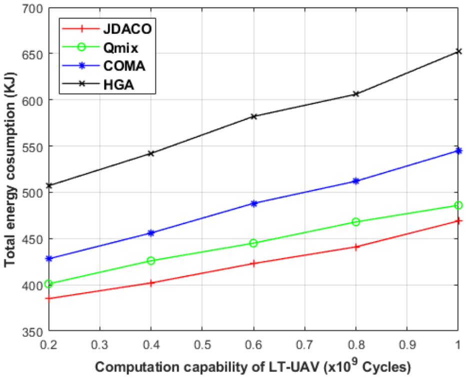
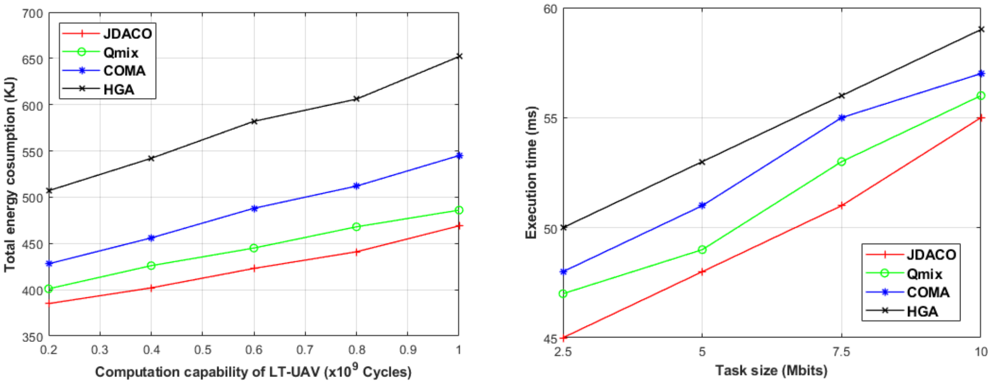


Figure 8. Total energy consumption of LT-UAV for different computation power of LT-UAV processor.



(a) (b)

Figure 9. Aggregation time (a) and execution time (b) for different task sizes

benchmarks. It should be noted that increasing task size influences the overall energy consumption for the process (i.e., the task) to be completed. Therefore, it is clear that increasing the task size increases the energy consumption of UAVs. As shown in **Figure. 10**, the proposed JDACO algorithm consumed the least amount of energy for the aggregation and computational offloading processes.

Similar to the energy consumption, the mission time by varying the task size is examined. Increasing the task size increases the overall mission time, because additional time is required to aggregate and compute the data for different task sizes. **Figure 11.** illustrates the different mission times required for the data aggregation and computation offloading processes for different schemes. The proposed JDACO scheme requires the least mission time compared to all the other benchmarks, as shown in **Figure 11**.

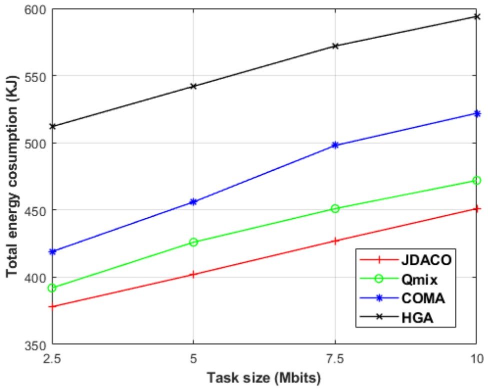


Figure 10. Total energy consumption of LT-UAVs for different task sizes.

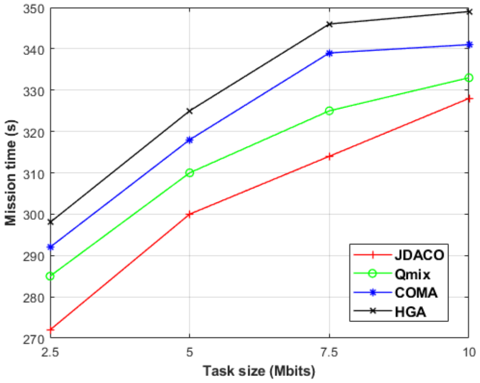


Figure 11. Mission time for different number of LT-UAVs.

6. Conclusion and Future Works

In this study, we presented a joint data aggregation and computation offloading scheme for post-disaster scenarios that minimizes the total cost of energy consumption and delays the aggregation and offloading processes. The joint optimization problem was defined and formulated as an MDP. We then solved the formulated the MDP problem by proposing an MA-DRL-based JDACO algorithm to perform discrete cooperative action. The simulation study shows that the proposed JDACO algorithm performs superiorly compared to other benchmarks in terms of mission execution time (i.e., delay) and energy consumption, while ensuring the maximum number of IoT devices in service.

In future work, not only consider mobile ground nodes will be considered, but also incorporation of the object detection ability of individual UAVs as an extension of this work. Additionally, we would like to further extend our work with the heterogeneous IoT nodes.

Bibliography

[1] Z. Wang, B. Liu, X. Cui, Y. Li, Y. Yang, and Z. Lyu, “The Application of Micro Coaxial Rotorcraft in Warfare: An Overview, Key Technologies, and Warfare Scenarios,” *IEEE Access*, vol. 10, pp. 40358–40366, 2022, doi: 10.1109/ACCESS.2022.3166890.

[2] N. H. Motlagh, M. Bagaa, and T. Taleb, “UAV-Based IoT Platform: A Crowd Surveillance Use Case,” *IEEE Commun. Mag.*, vol. 55, no. 2, pp. 128–134, 2017, doi: 10.1109/MCOM.2017.1600587CM.

[3] M. M. Alam and S. Moh, “Joint topology control and routing in a UAV swarm for crowd surveillance,” *J. Netw. Comput. Appl.*, vol. 204, no. May, p. 103427, 2022, doi: 10.1016/j.jnca.2022.103427.

[4] L. Chen, R. Zhao, K. He, Z. Zhao, and L. Fan, “Intelligent ubiquitous computing for future UAV-enabled MEC network systems,” *Cluster Comput.*, vol. 0123456789, 2021, doi: 10.1007/s10586-021-03434-w.

[5] V. Hassija *et al.*, “Fast, Reliable, and Secure Drone Communication: A Comprehensive Survey,” *IEEE Commun. Surv. Tutorials*, vol. 23, no. 4, pp. 2802–2832, 2021, doi: 10.1109/COMST.2021.3097916.

[6] M. Y. Arafat and S. Moh, “Location-Aided Delay Tolerant Routing Protocol in UAV Networks for Post-Disaster Operation,” *IEEE Access*, vol. 6, pp. 59891–59906, 2018, doi: 10.1109/ACCESS.2018.2875739.

[7] 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. Tutorials*, vol. 21, no. 3, pp. 2334–2360, 2019, doi: 10.1109/COMST.2019.2902862.

[8] C. You and R. Zhang, “Hybrid Offline-Online Design for UAV-Enabled Data Harvesting in Probabilistic LoS Channels,” *IEEE Trans. Wirel. Commun.*, vol. 19, no. 6, pp. 3753–3768, 2020, doi: 10.1109/TWC.2020.2978073.

[9] G. Geraci *et al.*, “What Will the Future of UAV Cellular Communications Be? A Flight from 5G to 6G,” *IEEE Commun. Surv. Tutorials*, vol. PP, p. 1, 2022, doi: 10.1109/COMST.2022.3171135.

[10] M. Y. Arafat and S. Moh, “Bio-inspired approaches for energy-efficient localization and clustering in uav networks for monitoring

wildfires in remote areas," *IEEE Access*, vol. 9, pp. 18649–18669, 2021, doi: 10.1109/ACCESS.2021.3053605.

[11] M. Y. Arafat and S. Moh, "A Q-Learning-Based Topology-Aware Routing Protocol for Flying Ad Hoc Networks," *IEEE Internet Things J.*, vol. 9, no. 3, pp. 1985–2000, 2022, doi: 10.1109/JIOT.2021.3089759.

[12] D. Wang, P. Hu, J. Du, P. Zhou, T. Deng, and M. Hu, "Routing and Scheduling for Hybrid Truck-Drone Collaborative Parcel Delivery With Independent and Truck-Carried Drones," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 10483–10495, 2019, doi: 10.1109/JIOT.2019.2939397.

[13] M. Y. Arafat and S. Moh, "JRCS: Joint Routing and Charging Strategy for Logistics Drones," *IEEE Internet Things J.*, vol. PP, pp. 1–1, 2022, doi: 10.1109/JIOT.2022.3182750.

[14] A. M. Raivi, S. M. A. Huda, M. M. Alam, and S. Moh, "Drone Routing for Drone-Based Delivery Systems: A Review of Trajectory Planning, Charging, and Security," *Sensors*, vol. 23, no. 3, p. 1463, Jan. 2023, doi: 10.3390/s23031463.

[15] Z. Zhang *et al.*, "Robust Semi-supervised Federated Learning for Images Automatic Recognition in Internet of Drones," *IEEE Internet Things J.*, vol. 4662, no. c, pp. 1–14, 2022, doi: 10.1109/JIOT.2022.3151945.

[16] H. Huang and A. V. Savkin, "Deployment of Heterogeneous UAV Base Stations for Optimal Quality of Coverage," *IEEE Internet Things J.*, vol. 4662, no. c, pp. 1–9, 2022, doi: 10.1109/JIOT.2022.3150292.

[17] V. Sy Nguyen, J. Jung, S. Jung, S. Joe, and B. Kim, "Deployable Hook Retrieval System for UAV Rescue and Delivery," *IEEE Access*, vol. 9, pp. 74632–74645, 2021, doi: 10.1109/ACCESS.2021.3080979.

[18] A. D. Boursianis *et al.*, "Internet of Things (IoT) and Agricultural Unmanned Aerial Vehicles (UAVs) in smart farming: A comprehensive review," *Internet of Things*, vol. 18, no. xxxx, p. 100187, May 2022, doi: 10.1016/j.iot.2020.100187.

[19] 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. Tutorials*, vol. 22, no. 2, pp. 1027–

1070, 2020, doi: 10.1109/COMST.2019.2962207.

[20] S. M. A. Huda and S. Moh, “Survey on computation offloading in UAV-Enabled mobile edge computing,” *J. Netw. Comput. Appl.*, vol. 201, no. February, p. 103341, 2022, doi: 10.1016/j.jnca.2022.103341.

[21] H. Wu, Z. Wei, Y. Hou, N. Zhang, and X. Tao, “Cell-Edge User Offloading via Flying UAV in Non-Uniform Heterogeneous Cellular Networks,” *IEEE Trans. Wirel. Commun.*, vol. 19, no. 4, pp. 2411–2426, 2020, doi: 10.1109/TWC.2020.2964656.

[22] S. Gu, Y. Wang, N. Wang, and W. Wu, “Intelligent Optimization of Availability and Communication Cost in Satellite-UAV Mobile Edge Caching System with Fault-Tolerant Codes,” *IEEE Trans. Cogn. Commun. Netw.*, vol. 6, no. 4, pp. 1230–1241, 2020, doi: 10.1109/TCCN.2020.3005921.

[23] W. Saad, M. Bennis, and M. Chen, “A Vision of 6G Wireless Systems: Applications, Trends, Technologies, and Open Research Problems,” *IEEE Netw.*, vol. 34, no. 3, pp. 134–142, 2020, doi: 10.1109/MNET.001.1900287.

[24] G. Geraci *et al.*, “What Will the Future of UAV Cellular Communications Be? A Flight from 5G to 6G,” *IEEE Commun. Surv. Tutorials*, vol. 24, no. 3, pp. 1304–1335, 2022, doi: 10.1109/COMST.2022.3171135.

[25] R. Akter, M. Golam, V.-S. Doan, J.-M. Lee, and D.-S. Kim, “IoMT-Net: Blockchain-Integrated Unauthorized UAV Localization Using Lightweight Convolution Neural Network for Internet of Military Things,” *IEEE Internet Things J.*, vol. 10, no. 8, pp. 6634–6651, Apr. 2023, doi: 10.1109/JIOT.2022.3176310.

[26] M. Dai, T. H. Luan, Z. Su, N. Zhang, Q. Xu, and R. Li, “Joint Channel Allocation and Data Delivery for UAV-Assisted Cooperative Transportation Communications in Post-Disaster Networks,” *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 9, pp. 16676–16689, 2022, doi: 10.1109/TITS.2022.3178789.

[27] “• IoT connected devices worldwide 2019-2030 | Statista.” https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/ (accessed Jun. 21, 2022).

[28] 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, doi: 10.1109/JIOT.2018.2887086.

[29] F. Zhou, R. Q. Hu, Z. Li, and Y. Wang, "Mobile edge computing in unmanned aerial vehicle networks," *IEEE Wirel. Commun.*, vol. 27, no. 1, pp. 140–146, 2020, doi: 10.1109/MWC.001.1800594.

[30] Q. V. Pham *et al.*, "Aerial Computing: A New Computing Paradigm, Applications, and Challenges," *IEEE Internet Things J.*, vol. 9, no. 11, pp. 8339–8363, 2022, doi: 10.1109/JIOT.2022.3160691.

[31] C. Zhan, H. Hu, X. Sui, Z. Liu, and D. Niyato, "Completion Time and Energy Optimization in the UAV-Enabled Mobile-Edge Computing System," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 7808–7822, 2020, doi: 10.1109/JIOT.2020.2993260.

[32] R. Han, L. Bai, J. Liu, J. Choi, and Y. C. Liang, "A Secure Structure for UAV-Aided IoT Networks: Space-Time Key," *IEEE Wirel. Commun.*, vol. 28, no. 5, pp. 96–101, 2021, doi: 10.1109/MWC.111.2100087.

[33] B. Alzahrani, O. S. Oubbati, A. Barnawi, M. Atiquzzaman, and D. Alghazzawi, "UAV assistance paradigm: State-of-the-art in applications and challenges," *J. Netw. Comput. Appl.*, vol. 166, no. February, p. 102706, Sep. 2020, doi: 10.1016/j.jnca.2020.102706.

[34] K. Messaoudi, O. S. Oubbati, A. Rachedi, A. Lakas, T. Bendouma, and N. Chaib, "A survey of UAV-based data collection: Challenges, solutions and future perspectives," *J. Netw. Comput. Appl.*, vol. 216, no. January, p. 103670, Jul. 2023, doi: 10.1016/j.jnca.2023.103670.

[35] P. McEnroe, S. Wang, and M. Liyanage, "A Survey on the Convergence of Edge Computing and AI for UAVs: Opportunities and Challenges," *IEEE Internet Things J.*, vol. 9, no. 17, pp. 15435–15459, 2022, doi: 10.1109/JIOT.2022.3176400.

[36] A. M. Raivi and S. Moh, "Vision-Based Navigation for Urban Air Mobility : A Survey," in *Proc. of 12th Int. Conf. on Smart Media and Applications (SMA 2023)*, 2023, pp. 1–6.

[37] A. M. Raivi and S. Moh, "Trajectory Planning Techniques in Urban Air Mobility: A Comparative Survey," *Proc. 11th Int. Conf. Smart Media Appl. (SMA 2022)*, no. October, pp. 15–20, 2022.

[38] K. Li, W. Ni, A. Noor, and M. Guizani, “Employing Intelligent Aerial Data Aggregators for the Internet of Things: Challenges and Solutions,” *IEEE Internet Things Mag.*, vol. 5, no. 1, pp. 136–141, 2022, doi: 10.1109/iotm.001.2100161.

[39] S. M. A. Huda and S. Moh, “Survey on computation offloading in UAV-Enabled mobile edge computing,” *J. Netw. Comput. Appl.*, vol. 201, no. October 2021, p. 103341, 2022, doi: 10.1016/j.jnca.2022.103341.

[40] A. Bera, S. Misra, C. Chatterjee, and S. Mao, “CEDAN: Cost-Effective Data Aggregation for UAV-Enabled IoT Networks,” *IEEE Trans. Mob. Comput.*, vol. 1233, no. c, pp. 1–1, 2022, doi: 10.1109/TMC.2022.3172444.

[41] A. A. Baktayan and I. A. Al-Baltah, “A survey on intelligent computation offloading and pricing strategy in UAV-Enabled MEC network: Challenges and research directions,” *Sustain. Eng. Innov.*, vol. 4, no. 2, pp. 156–190, Dec. 2022, doi: 10.37868/sei.v4i2.id179.

[42] A. M. Raivi and S. Moh, “A comprehensive survey on data aggregation techniques in UAV-enabled Internet of things,” *Comput. Sci. Rev.*, vol. 50, no. 2, p. 100599, Nov. 2023, doi: 10.1016/j.cosrev.2023.100599.

[43] K. Li, W. Ni, A. Noor, and M. Guizani, “Employing Intelligent Aerial Data Aggregators for the Internet of Things: Challenges and Solutions,” *IEEE Internet Things Mag.*, vol. 5, no. 1, pp. 136–141, Mar. 2022, doi: 10.1109/IOTM.001.2100161.

[44] X. Wang, J. Hu, and H. Lin, “An intelligent UAV based data aggregation strategy for IoT after disaster scenarios,” *DroneCom 2020 - Proc. 2nd ACM MobiCom Work. Drone Assist. Wirel. Commun. 5G Beyond*, pp. 97–101, 2020, doi: 10.1145/3414045.3415940.

[45] A. Islam, A. Al Amin, and S. Y. Shin, “FBI: A Federated Learning-Based Blockchain-Embedded Data Accumulation Scheme Using Drones for Internet of Things,” *IEEE Wirel. Commun. Lett.*, vol. 11, no. 5, pp. 972–976, 2022, doi: 10.1109/LWC.2022.3151873.

[46] G. Hattab and D. Cabric, “Energy-Efficient Massive IoT Shared Spectrum Access over UAV-Enabled Cellular Networks,” *IEEE Trans. Commun.*, vol. 68, no. 9, pp. 5633–5648, 2020, doi:

10.1109/TCOMM.2020.2998547.

[47] A. A. Al-Habob, O. A. Dobre, and H. Vincent Poor, “Energy-efficient spatially-correlated data aggregation using unmanned aerial vehicles,” *IEEE Int. Symp. Pers. Indoor Mob. Radio Commun. PIMRC*, vol. 2020-Augus, 2020, doi: 10.1109/PIMRC48278.2020.9217066.

[48] T. D. Nguyen, D. T. Le, N. Pham-Van, H. Choo, and T. Pham Van, “UAV-aided Sensory Data Aggregation: Incremental Clustering and Scheduling Approach,” *Proc. 2021 15th Int. Conf. Ubiquitous Inf. Manag. Commun. IMCOM 2021*, 2021, doi: 10.1109/IMCOM51814.2021.9377438.

[49] O. M. Bushnaq, A. Celik, H. Elsawy, M. S. Alouini, and T. Y. Al-Naffouri, “Aeronautical Data Aggregation and Field Estimation in IoT Networks: Hovering and Traveling Time Dilemma of UAVs,” *IEEE Trans. Wirel. Commun.*, vol. 18, no. 10, pp. 4620–4635, 2019, doi: 10.1109/TWC.2019.2921955.

[50] L. Bai, J. Liu, J. Wang, R. Han, and J. Choi, “Data Aggregation in UAV-Aided Random Access for Internet of Vehicles,” *IEEE Internet Things J.*, vol. 9, no. 8, pp. 5755–5764, 2022, doi: 10.1109/JIOT.2021.3063734.

[51] K. Li, W. Ni, Y. Emami, and F. Dressler, “Data-Driven Flight Control of Internet-of-Drones for Sensor Data Aggregation Using Multi-Agent Deep Reinforcement Learning,” *IEEE Wirel. Commun.*, vol. 29, no. 4, pp. 18–23, Aug. 2022, doi: 10.1109/MWC.002.2100681.

[52] S. M. A. Huda and S. Moh, “Deep Reinforcement Learning-Based Computation Offloading in UAV Swarm-Enabled Edge Computing for Surveillance Applications,” *IEEE Access*, vol. 11, no. June, pp. 68269–68285, 2023, doi: 10.1109/ACCESS.2023.3292938.

[53] Z. Jia, Q. Wu, S. Member, and C. Dong, “Hierarchical Aerial Computing for Internet of Things via Cooperation of HAPs and UAVs,” *IEEE Internet Things J.*, vol. 10, no. 7, pp. 5676–5688, 2023, doi: 10.1109/JIOT.2022.3151639.

[54] F. Song *et al.*, “Evolutionary Multi-Objective Reinforcement Learning Based Trajectory Control and Task Offloading in UAV-Assisted Mobile Edge Computing,” *IEEE Trans. Mob. Comput.*, pp. 1–18, 2022, doi: 10.1109/TMC.2022.3208457.

[55] T. Cai *et al.*, “Cooperative Data Sensing and Computation Offloading in UAV-Assisted Crowdsensing With Multi-Agent Deep Reinforcement Learning,” *IEEE Trans. Netw. Sci. Eng.*, vol. 9, no. 5, pp. 3197–3211, Sep. 2022, doi: 10.1109/TNSE.2021.3121690.

[56] Y. Yu, J. Tang, J. Huang, X. Zhang, D. K. C. So, and K.-K. Wong, “Multi-Objective Optimization for UAV-Assisted Wireless Powered IoT Networks Based on Extended DDPG Algorithm,” *IEEE Trans. Commun.*, vol. 69, no. 9, pp. 6361–6374, Sep. 2021, doi: 10.1109/TCOMM.2021.3089476.

[57] Y. Liu, J. Yan, and X. Zhao, “Deep Reinforcement Learning Based Latency Minimization for Mobile Edge Computing With Virtualization in Maritime UAV Communication Network,” *IEEE Trans. Veh. Technol.*, vol. 71, no. 4, pp. 4225–4236, Apr. 2022, doi: 10.1109/TVT.2022.3141799.

[58] Q. Liu, H. Liang, R. Luo, and Q. Liu, “Energy-Efficiency Computation Offloading Strategy in UAV Aided V2X Network With Integrated Sensing and Communication,” *IEEE Open J. Commun. Soc.*, vol. 3, no. June, pp. 1337–1346, 2022, doi: 10.1109/OJCOMS.2022.3195703.

[59] C. E. Miller, R. A. Zemlin, and A. W. Tucker, “Integer Programming Formulation of Traveling Salesman Problems,” *J. ACM*, vol. 7, no. 4, pp. 326–329, 1960, doi: 10.1145/321043.321046.

[60] R. Han, H. Li, E. J. Knoblock, M. R. Gasper, and R. D. Apaza, “Joint Velocity and Spectrum Optimization in Urban Air Transportation System via Multi-agent Deep Reinforcement Learning,” *IEEE Trans. Veh. Technol.*, pp. 1–13, 2023, doi: 10.1109/TVT.2023.3256067.

[61] M. L. Littman, “Markov games as a framework for multi-agent reinforcement learning,” in *Machine Learning Proceedings 1994*, vol. 120, no. 1, Elsevier, 1994, pp. 157–163. doi: 10.1016/B978-1-55860-335-6.50027-1.

[62] P. Sunehag *et al.*, “Value-Decomposition Networks For Cooperative Multi-Agent Learning,” *Proc. Int. Jt. Conf. Auton. Agents Multiagent Syst. AAMAS*, vol. 3, pp. 2085–2087, Jun. 2017, [Online]. Available: http://arxiv.org/abs/1706.05296

[63] A. G. B. Richard S. Sutton, *Reinforcement Learning: An Introduction,*

Second Edi. A Bradford Book, 1998.

[64] Z. Wang, T. Schaul, M. Hessel, H. Van Hasselt, M. Lanctot, and N. De Frcitas, “Dueling Network Architectures for Deep Reinforcement Learning,” *33rd Int. Conf. Mach. Learn. ICML 2016*, vol. 4, no. 9, pp. 2939–2947, 2016.

[65] B. Jang, M. Kim, G. Harerimana, and J. W. Kim, “Q-Learning Algorithms: A Comprehensive Classification and Applications,” *IEEE Access*, vol. 7, pp. 133653–133667, 2019, doi: 10.1109/ACCESS.2019.2941229.

[66] H. Van Hasselt, A. Guez, and D. Silver, “Deep reinforcement learning with double Q-Learning,” *30th AAAI Conf. Artif. Intell. AAAI 2016*, pp. 2094–2100, 2016, doi: 10.1609/aaai.v30i1.10295.

[67] V. Mnih *et al.*, “Human-level control through deep reinforcement learning,” *Nature*, vol. 518, no. 7540, pp. 529–533, 2015, doi: 10.1038/nature14236.

[68] T. Rashid, M. Samvelyan, C. S. de Witt, G. Farquhar, J. Foerster, and S. Whiteson, “QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning,” *35th Int. Conf. Mach. Learn. ICML 2018*, vol. 10, pp. 4295–4304, Mar. 2018, [Online]. Available: http://arxiv.org/abs/2003.08839

[69] J. N. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, “Counterfactual multi-agent policy gradients,” *32nd AAAI Conf. Artif. Intell. AAAI 2018*, pp. 2974–2982, 2018, doi: 10.1609/aaai.v32i1.11794.

[70] S. Mitchell, M. O’Sullivan, and I. Dunning, “PuLP: A Linear Programming Toolkit for Python,” *Univ. Auckland, Auckland, New Zeal.*, p. 65, 2011, [Online]. Available: http://www.optimization-online.org/DB_FILE/2011/09/3178.pdf

[71] Y. Wang *et al.*, “Trajectory Design for UAV-Based Internet of Things Data Collection: A Deep Reinforcement Learning Approach,” *IEEE Internet Things J.*, vol. 9, no. 5, pp. 3899–3912, 2022, doi: 10.1109/JIOT.2021.3102185.

[72] A. M. Raivi and S. Moh, “JDACO: Joint Data Aggregation and Offloading in UAV-Enabled Internet of Things for Post-Disaster Scenario,” Submitted to *IEEE Internet of Things J.*, Sep. 2023.

Acknowledgement

I am deeply grateful to all those who have supported me throughout this journey, without whom this thesis would not have been possible.

First and foremost, I would like to express my profound and sincere gratitude to my advisor, Prof. Sangman Moh. His invaluable support, encouragement, supervision, and useful suggestions throughout the master's studies have provided strong base for the completion of my thesis. I would always be indebted to him for his quintessential professionalism, persistent guidelines, and continuous encouragement through his impeccable mentorship.

Secondly, I want to express my warm and sincere thanks to the thesis committee members, Prof. Seokjoo Shin and Prof. Moonsoo Kang for their constructive comments and invigorating suggestions. All their insights regarding my research works have helped me in improving and extending it in different ways.

My sincere acknowledgement to the Department of Computer Engineering and Mobile Computing Lab for providing me such a wonderful opportunity and an atmosphere to grow me academically and otherwise. I must be thankful to all my lab members for their moral as well as academic support. I would like to heartily thank all my seniors and friends from society of Bangladesh in Chosun University for their affection and cooperation that made my life easier and cheerful in Korea.

At last, but not the least, I cannot stop myself from thanking my parents, my lovely wife Auditi and remaining family members. Without their encouragement and love, it would have been impossible for me achieve anything. I would like to dedicate my work to them.