drones

MDPI

Article

DELOFF: Decentralized Learning-Based Task Offloading for Multi-UAVs in U2X-Assisted Heterogeneous Networks

Anqi Zhu [1], Huimin Lu [1], Mingfang Ma [2,3], Zongtan Zhou [1] and Zhiwen Zeng [1,*]

[1] College of Intelligence Science and Technology, National University of Defense Technology, Changsha 410073, China; anqi@nudt.edu.cn (A.Z.); lhmnew@nudt.edu.cn (H.L.); ztzhou@nudt.edu.cn (Z.Z.)
[2] College of Science, National University of Defense Technology, Changsha 410073, China; mmf@nudt.edu.cn
[3] Yanken School, Yancheng 224100, China
* Correspondence: zengzhiwen@nudt.edu.cn

Abstract: With multi-sensors embedded, flexible unmanned aerial vehicles (UAVs) can collect sensory data and provide various services for all walks of life. However, limited computing capability and battery energy put a great burden on UAVs to handle emerging compute-intensive applications, necessitating them to resort to innovative computation offloading technique to guarantee quality of service. Existing research mainly focuses on solving the offloading problem under known global information, or applying centralized offloading frameworks when facing dynamic environments. Yet, the maneuverability of today's UAVs, their large-scale clustering, and their increasing operation in the environment with unrevealed information pose huge challenges to previous work. In this paper, in order to enhance the long-term offloading performance and scalability for multi-UAVs, we develop a decentralized offloading scheme named DELOFF with the support of mobile edge computing (MEC). DELOFF considers the information uncertainty caused by the dynamic environment, uses UAV-to-everything (U2X)-assisted heterogeneous networks to extend network resources and offloading flexibility, and tackles the joint strategy making related to computation mode, network selection, and offloading allocation for multi-UAVs. Specifically, the optimization problem of multi-UAVs is addressed by the proposed offloading algorithm based on a multi-arm bandit learning model, where each UAV itself can adaptively assess the offloading link quality through the fuzzy logic-based pre-screening mechanism designed. The convergence and effectiveness of the DELOFF proposed are also demonstrated in simulations. And, the results confirm that DELOFF is superior to the four benchmarks in many respects, such as reduced consumed energy and delay in the task completion of UAVs.

Keywords: unmanned aerial vehicle; computation offloading; heterogeneous networks; multi-arm bandit; fuzzy logic

Citation: Zhu, A.; Lu, H.; Ma, M.; Zhou, Z.; Zeng, Z. DELOFF: Decentralized Learning-Based Task Offloading for Multi-UAVs in U2X-Assisted Heterogeneous Networks. *Drones* **2023**, *7*, 656. https://doi.org/10.3390/drones7110656

Academic Editor: Emmanouel T. Michailidis

Received: 13 September 2023
Revised: 26 October 2023
Accepted: 28 October 2023
Published: 1 November 2023

1. Introduction

With the popularization of the Internet of Things (IoT), massive applications brought by the proliferation of intelligent terminals are emerging, changing people's production and lifestyles [1,2]. As typical use cases, cutting-edge applications of popular unmanned aerial vehicles (UAVs) driven by IoT play an indispensable role in transportation systems [3], agriculture 4.0 [4], environmental monitoring [5], etc. As a result, the computational requirements for processing these novel applications are rapidly increasing [6]. However, to maintain lightweight and maneuverability properties, most UAVs are limited by their local computation and battery capacity, and in most cases, they cannot effectively compute for these latency-sensitive IoT applications [7]. Thus, how to utilize external network resources to process computation tasks efficiently and in a timely manner for intelligent mobile UAVs is a critical challenge that needs to be addressed urgently.

Fortunately, computation offloading as a novel technique is proposed, which can effectively support the applications of UAVs and assist in processing the tasks generated [8].

Several studies are devoted to offloading and recommend task offloading from limited terminals to remote cloud or edge servers. Particularly, mobile edge computing (MEC), as a novel paradigm, has received extensive attention [9,10]. This can provide powerful external computing capability to UAVs at the network edge, which improves the offloading efficiency [11]. For example, by using computation offloading, Liu et al. [12] proposed a delay optimization approach to reduce the computing delay greatly. And, Bacanin et al. [13] designed an offloading method dedicated to optimizing energy to minimize the energy consumed during task computation effectively.

These works alleviate, to some extent, the performance issues caused by the limited endurance and restricted computational capabilities of UAVs. Still, they only focus on offloading tasks over a single network, which ignores the potential advantages of current heterogeneous network architecture and limits the available network options for UAV offloading, hindering the flexibility and adaptability of flying UAVs in heterogeneous network environments. In addition, most existing approaches rely on offloading tasks to fixed servers, which may introduce latency and dependency on a central infrastructure. This can create significant pitfalls when dealing with latency-sensitive applications or in situations when the server becomes inaccessible or experiences failures.

Therefore, in this paper, to support massive access and service requests, we consider the impact of network heterogeneity on offloading performance. The popular 5G introduces and deploys different radio access technologies, including cellular base stations (BSs) and WiFi access points (APs) [14,15]. By allowing UAVs to offload computation tasks over heterogeneous networks, UAVs can take advantage of different network options. Furthermore, differently from previous works, we consider the original UAV-to-everything (U2X) paradigm that enables UAVs to offload tasks to various entities, including other UAVs, ground-based vehicles, infrastructures, and IoT devices [16]. The popularity of UAVs and other smart terminals make it possible to offload among terminals, which will effectively reduce the risk of network congestion in data-intensive areas, and extend the service coverage of edge network in remote areas with few infrastructures. This enhances the flexibility and adaptability of UAV offloading, resulting in improved performance and reliability.

However, in the proposed U2X-assisted MEC scenario, there are some key issues that have not been effectively researched, which pose significant challenges to optimizing the offloading performance of UAVs, including:

(a) Unstable offloading links: Differently from previous studies in which servers are fixed by default and devices are assumed to be stationary, in U2X networks, offloading links may not be always stable due to the movement of UAVs or even offloading entities. This may cause link disconnections, frequent network switching and even ping-pong effects, greatly affecting offloading performance.
(b) Centralized offloading frameworks: Much of the existing work relies on centralized frameworks to address the offloading problem, which may not be suitable for UAV systems in such a scenario. Additionally, centralized approaches can introduce communication overhead, scalability issues, and a single point of failure.
(c) Environmental information uncertainty: Most existing work often assumes that information about the environment is all known in advance. However, in real-world scenarios, this assumption is unrealistic due to the large-scale network environment, which makes it difficult for UAVs to obtain all of the information. The uncertainty of global state information limits the applicability of previous solutions, posing brand-new challenges to the effectiveness of UAV offloading.

In response to the aforementioned challenges, this paper formulates the optimization problem of UAV offloading in heterogeneous networks as the utility maximization for multi-UAVs through making the joint strategy related to the computation mode, network selection, and offloading allocation, and then proposes a decentralized offloading scheme using a U2X-assisted MEC named DELOFF to solve it. The DELOFF proposed achieves the following aims, including:

(1) Offloading quality enhancement: By perceiving the motion state of the UAV and the quality of wireless links approximately, a fuzzy logic-based pre-screening mechanism is devised and executed on the UAV side, so as to assess and identify the potential stable offloading nodes dynamically in U2X networks.
(2) Decentralized offloading framework: The offloading problem for multi-UAVs is transformed into a multi-arm bandit (MAB) learning model in the paper, where the UAV plays the role of agent, and the arm machine is the strategy of the UAV for task offloading allocation. And, DELOFF can adaptively acquire the optimal offloading solution by relying on UAVs to make task offloading decisions locally in a decentralized manner.
(3) Offloading under information uncertainty: By balancing the exploration–exploitation trade-off and utilizing bandit feedback to derive an offloading strategy that can achieve higher potential rewards, DELOFF dynamically adjusts the computation offloading optimization for multi-UAVs in a decentralized manner, and UAVs do not need to communicate with each other for certain information.

Overall, our contributions are summarized as follows:

- Unlike the previous works that consider the scene with only one network or fixed offloading server, we introduce a UAV offloading model in the U2X-assisted heterogeneous networks to improve the efficiency and flexibility of offloading. Furthermore, to avoid additional centralized management that increases overhead, we propose a decentralized offloading scheme named DELOFF, which can acquire the optimal solution by leveraging the feedback of empirical reward of task offloading.
- To adaptively evaluate potential offloaded nodes in the dynamic U2X-assisted scenario, a lightweight pre-screening mechanism is designed to perform on the UAV. The mechanism can guarantee stable offloading of mobile UAVs and evaluate service nodes at the risk of disconnection or poor offloading quality such as unexpected packet loss rate (PLR), without extra demand for information interaction between UAVs and numerous nodes over heterogeneous networks.
- By considering the uncertainties associated with UAV mobility and environment, the DELOFF proposed decouples the learning process into exploration, assignment, and exploitation, further coping effectively with the dynamic offloading and enabling more robust strategy making for UAVs. Further, the UAV swarm can be scaled up at will, and the UAVs no longer need to communicate internally or be aware of the presence of each other in our model, which has great scalability.
- We implement simulations to verify the rationality and effectiveness of the DELOFF proposed. The results demonstrate that DELOFF can achieve steady convergence, and show superiority in various scenarios compared to the other four benchmarks.

The rest of the paper is organized as below. Section 2 overviews the related work. In Section 3, we depict the system model, introduce the task computation models, and further formulate the optimization problem of task offloading allocation for multi-UAVs. Moreover, Section 4 shows the proposed offloading pre-screening mechanism, and Section 5 elaborates the details of the developed DELOFF scheme. Finally, Section 6 implements simulations to demonstrate the DELOFF and Section 7 concludes this paper.

2. Related Work

Nowadays, more and more UAVs can access the infrastructures wirelessly. The MEC paradigm offers more possibilities for the implementation of novel computation-intensive or delay-sensitive applications for UAVs at the edge. Effective task offloading through MEC can ease resource constraints on UAVs and support the requirements of applications [17]. As a result, a series of research on computation offloading has been carried out, and we briefly outline the related work.

Some studies consider centralized controllers to achieve offloading decisions. Li et al. [18] considered that the tasks executed by UAVs have demands for delay, and exploited a genetic-based offloading scheme for the energy-limited UAVs, which optimizes

the consumed energy under delay constraint. Alfakih et al. [19] designed an offloading algorithm based on reinforcement learning to make appropriate offloading decisions and minimize system costs. These methods are poorly adapted to non-centralized computing infrastructures [20]. And, they require complete information collection for offloading, which further increases the delay. In addition, the offloading approaches proposed in [21–24] through centralized manners are difficult to manage and control when the system scale expands continuously, and they also have undesirable scalability and robustness.

Therefore, some research is carried out in a distributed architecture. Dai et al. [25] designed a matching-based offloading framework for UAVs in smart cities, which can utilize resources at the network edge and improve offloading efficiency for UAVs. Xu et al. [26] used a distributed game approach to accomplish the computation offloading, balancing the benefits of satisfying the application and relieving the pressure on the server. Nguyen et al. [27] designed a UAV offloading scheme to improve task completion time by virtue of reinforcement learning, in which each UAV can only make strategies independently and offload to the fixed MEC server without sharing information and potential cooperation. However, these studies [25–29] only use a single network, ignoring the complementarity of hybrid network resources to single cellular resources in heterogeneous network environments. Therefore, some studies have tried to consider the effect of network heterogeneity in offloading. In the reference [30,31], the offloading mechanisms proposed all consider preferential offloading through WiFi AP when WiFi is available; otherwise, the offloading will take place over the cellular BS. These mechanisms are carried out through a fixed network selection mode, which can cause wastage of cellular resources accompanied by WiFi transmission blockage due to the lack of flexibility.

In addition, existing offloading studies [32,33] ignore the idle resources in neighbors and the assumed environment is mostly static, without taking the uncertain and stochastic dynamic environment into account. In fact, UAVs need shared edge resources to process tasks, but they are unclear with both the information of the system and the demands of other UAVs. In addition, to avoid the potential risks caused by poor transmission, it cannot be overlooked how to evaluate each available service node and establish an appropriate offloading link for mobile UAVs in this densely covered network environment, which is also a key issue that is not considered in previous studies.

In this paper, an offloading mechanism is considered to make appropriate decisions for multi-UAVs with different motion states in uncertain environments, which has a joint measure of service capability, velocity, and transmission link quality. Furthermore, to achieve decentralized offloading for numerous coexisting UAVs in such stochastic and complex environments, a learning-based adaptive approach is designed, in which heterogeneous applications vary in demands, and the exploration–exploitation trade-off is well balanced.

3. System Model and Problem Formulation

3.1. System Model

We consider a U2X-assisted MEC system in heterogeneous networks, as shown in Figure 1, where the fixed APs and BSs are spatially distributed and fitted with a set of MEC servers. Within the available service capacity of the servers, they can provide radio access and computing services for the computational intensive or delay sensitive application tasks generated by the flying UAVs in the radio range. In addition, considering the availability of the common edge infrastructures, any widely deployed smart terminals in the scenario such as vehicles and UAVs can act as offloading nodes or service helper terminals to help perform certain services for the UAVs within their communication range.

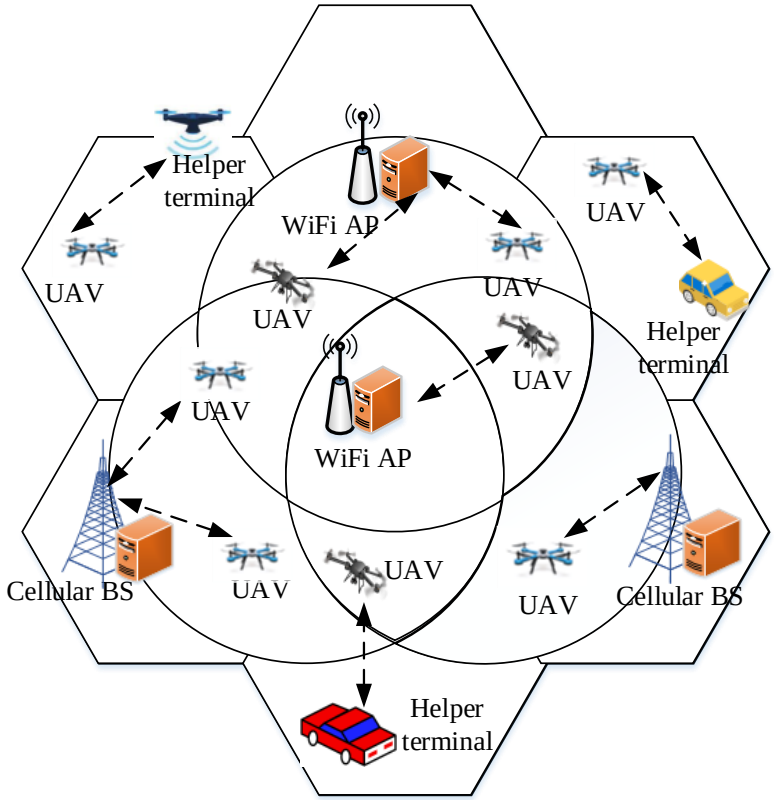


Figure 1. Multi-UAV task offloading in U2X-assisted heterogeneous networks.

There is a set of UAVs $\mathcal{H} = \{1, ..., H\}$, and N servers, which is collected in a set $\mathcal{N} = \{1, ..., N\}$, and let the set of smart terminals be $\mathcal{M} = \{1, ..., M\}$. For ease of analysis, we refer to all edge offloading nodes in the system as the set $\mathcal{S}$, $\mathcal{S} = \mathcal{N} \cup \mathcal{M}$. The task from UAV h is typically expressed by a tuple $\psi_h = \{d_{\psi_h}, c_{\psi_h}\}$, in which d_{ψ_h} and c_{ψ_h} represent the data size and the computation amount required to accomplish task ψ_h, respectively. UAVs have differentiated quality of service (QoS) requirements that may vary for diverse application requests. Specifically, to ensure satisfactory QoS support, the maximum thresholds tolerable to the critical attributes, including delay and PLR, are considered when evaluating the normal execution for the applications, which are defined as d_{max} and p_{max}, respectively.

We further consider a slotted scenario for the system, where each UAV $h \in \mathcal{H}$ generates a computing task at any time slot $t \in \{1, 2, ..., T\}$ and can choose to process the task locally, offload through heterogeneous RATs to execute at an edger server, or perform the task by transferring to a smart terminal through U2X communication. Within the entire time period, the above different strategies for processing a computation task are accordingly represented using variable a_h. Specifically, $a_h = 0$ indicates that local computing is adopted; $a_h = 1$ represents that the UAV chooses to offload the task to an edge server $n \in \mathcal{N}$; and $a_h = 2$ signifies that the task is executed at a terminal $m\mathcal{M}$. As a result, these task offloading allocation strategies that UAV h can adopt are included in a set $A = \{a_h | a_h = 0, 1, 2\}$, and each computation task ψ_h can finally be processed by, at most, one of the strategies.

The system information associated with the CPU processing speed of each edge node $s \in \mathcal{S}$ is usually undisclosed to UAVs. Moreover, the communication links between UAVs and edge nodes may be affected by various factors, such as network congestion, signal interference, etc. These factors lead to changes and instability in transmission rates, which may vary even within the same time period. Consequently, the uncertainty and variability of system information including the processing speed and transmission rate pose significant challenges for UAVs when making network selection and offloading strategies, which facilitates us to devise a decentralized task offloading mechanism without involving information interchange, where all UAVs make offloading strategies locally.

To facilitate easy referencing, Table 1 provides the definitions for the essential symbols.

Table 1. Key Symbol Definitions.

Symbols	Definition
$\mathcal{H} = \{1, ..., H\}$	The set of UAVs
$\mathcal{N} = \{1, ..., N\}$	The set of servers
$\mathcal{M} = \{1, ..., M\}$	The set of helper terminals
$\mathcal{S} = \mathcal{N} \cup \mathcal{M}$	The set of edge offloading nodes
$\psi_h = \{d_{\psi_h}, c_{\psi_h}\}$	The computation task of each UAV $h \in \mathcal{H}$
d_{ψ_h}	The size of task data
c_{ψ_h}	The required computation amount to accomplish task
$t \in \{1, 2, ..., T\}$	The time slot
$A = \{a_h \| a_h = 0, 1, 2\}$	The set of offloading strategies
$f_{uav,h}$	The computation capability of UAV h
$f_{uav,n}$	The computation capability of server n
$f_{uav,m}$	The computation capability of terminal m
$d_{\psi_h}^{loc,exe}$	The local computing time of UAV h
$e_{\psi_h}^{loc,exe}$	The energy consumption of UAV h in local computing
$d_{\psi_h,n}^{tr}$	The transmission time from UAV h to terminal m
$d_{\psi_h,n}^{exe}$	The task execution time on server n
$e_{\psi_h,n}^{tr}$	The energy consumption of UAV h for transmitting data to server n
γ_h^0	The energy coefficient of UAV h
$d_{\psi_h,m}^{tr}$	The task transmission time from UAV h to terminal m
$e_{\psi_h,m}^{tr}$	The energy consumption of UAV h for transmitting task data to terminal m
$d_{\psi_h,m}^{exe}$	The task execution time on terminal n
B^c	The channel bandwidth for cellular network
P_h^c	The transmission power of UAV via cellular link
Dr_h^c	The data transmission rate from UAV h to server n via cellular link
$G_{h,n}^c$	The channel gain from UAV h to the BS
$loc_h = [x_h, y_h, z_h]$	The position coordinates of UAV h
$loc_n = [x_n, y_n]$	The position coordinates of server n
$(\sigma^c)^2$	The variance associated with the additive white Gaussian noise
B^w	The channel bandwidth for WiFi link
Dr_h^w	The data transmission rate from UAV h to server n via WiFi link
P_h^w	The transmission power of UAV via WiFi link
$G_{h,n}^w$	The channel gain from UAV h to the WiFi AP
$e_{\psi_h}^{hover}$	The hovering energy consumption of UAV h
P_{hover}	The hovering power of UAV h
$d_{total}^{\psi_h}$	The time of UAV for computation offloading under different strategies
$e_{total}^{\psi_h}$	The energy consumption of UAV under different offloading strategies
Dr_h^m	The data transmission rate of the wireless link from UAV h to terminal m
$B_{h,m}$	The bandwidth of the wireless link between UAV h and terminal m
$P_{h,m}$	The transmission power for offloading task from UAV h to terminal m
$G_{h,m}$	The channel gain of the wireless link between UAV h and terminal m
u_{h,a_h}	The utility of UAV for processing computation task ψ_h
$\delta_{\psi_h,d}$, $\delta_{\psi_h,e}$	The delay and energy weight factors

3.2. Multi-Mode Task Computation

(1) Local Computing: Let us define $f_{uav,h}$ as the computation capability of UAV h, representing the CPU chip's clock frequency (i.e., G cycles/s), and denote c_{ψ_h} as the required computation resources (i.e., G cycles) to finish task ψ_h of UAV h, which indicates the total number of clock cycles to compute the task. UAVs can flexibly consider the use of CPUs or

GPUs to perform task computations, depending on the computational requirements of the specific task [34].

Therefore, when task ψ_h is computed by UAV h locally, the computation execution time can be given by

$$d_{\psi_h}^{loc,exe} = \frac{c_{\psi_h}}{f_{uav,h}}, \ \forall h \in \mathcal{H} \tag{1}$$

The energy consumption of UAV h in local computing is mainly dominated by the CPU computation process. Let $e_{\psi_h}^{loc,exe}$ define the energy consumption of UAV h if its task ψ_h is computed locally, which can be calculated as in [35]:

$$e_{\psi_h}^{loc,exe} = \gamma_h^0 f_{uav,h}^3 \cdot d_{\psi_h}^{loc,exe} = \gamma_h^0 f_{uav,h}^2 \cdot c_{\psi_h}, \ \forall h \in \mathcal{H} \tag{2}$$

where γ_h^0 indicates the energy coefficient related to the CPU chip architecture of UAV h, and $\gamma_h^0 = 10^{-26}$. $\gamma_h^0 f_{uav,h}^3$ expresses the computation power of UAV h. Further, given the local computing time $d_{\psi_h}^{loc,exe}$ according to Equation (1), the energy consumption of UAV h for executing the task locally is further expressed as $\gamma_h^0 f_{uav,h}^2 \cdot c_{\psi_h}$.

(2) MEC Offloading: The computation task data can be transmitted from UAVs to edge nodes via wireless channel. As previously mentioned, the AP in wireless communication can be either cellular BS or WiFi AP. The channel from the flying UAV $h \in \mathcal{H}$ to an AP follows quasi-static block fading [36,37]. By using the Shannon formula, the expected data transmission rate of UAV h for offloading the task ψ_h via the orthogonal channel of the cellular BS to an edge server $n \in \mathcal{N}$ can be represented as [38,39]

$$Dr_h^c = B^c \cdot \log_2(1 + \frac{P_h^c G_{h,n}^c}{(\sigma^c)^2}) \tag{3}$$

where B^c denotes the channel bandwidth for cellular access, the transmission power of UAV via cellular link is P_h^c, $(\sigma^c)^2$ indicates the variance associated with the additive white Gaussian noise, and $G_{h,n}^c$ is the channel gain from UAV h to the BS n when offloading task ψ_h, $G_{h,n}^c = \lambda_{h,n}^c |\chi_{h,n}|^2$. Further, $\chi_{h,n}$ represents a small-scale fading coefficient, which is a random variable that follows the Gaussian distribution, i.e., $\chi_{h,n} \sim \mathcal{CN}(0,1)$; on the other side, according to theoretical analysis and empirical measurement, the path loss model for the large-scale fading gain can be represented as $\lambda_{h,n}^c = 128.1 + 37.6 \log_{10} l_{h,n}(dB)$ [40], in which $l_{h,n}$ is the transmission distance between UAV h and server n. We define the position of UAV $h \in \mathcal{H}$ and server $n \in \mathcal{N}$ as $loc_h = [x_h, y_h, z_h]$ and $loc_n = [x_n, y_n]$, respectively, thereby the distance is calculated by $l_{h,n} = \sqrt{|z_h|^2 + \|[x_h, y_n] - [x_h, y_n]\|^2}$.

Likewise, the data rate for offloading task over WiFi connectivity can be expressed as

$$Dr_h^w = B^w \cdot \log_2(1 + \frac{P_h^w G_{h,n}^w}{(\sigma^w)^2}) \tag{4}$$

Similar to many studies such as [36], the variables in Equation (4) have the same meaning as those in Equation (3). However, there are distinct differences in the values of the channel bandwidth and path loss model for the two communication technologies when estimating the transmission rate. Particularly, in Equation (4), $G_{h,n}^w = \lambda_{h,n}^w |\chi_{h,n}|^2$, where the path loss model for the large-scale fading gain is calculated as $\lambda_{h,n}^c = 89.5 + 10 \log_{10} l_{h,n}(dB)$.

Based on Equations (3) and (4), let $Dr_h \in \{Dr_h^c, Dr_h^w\}$ be the data rate for the offloading task from UAV h to a server $n \in \mathcal{N}$ through cellular or WiFi wireless link. Accordingly, the transmission time $d_{\psi_h,n}^{tr}$ of the wireless link between UAV h and server n can be expressed by

$$d_{\psi_h,n}^{tr} = \frac{d_{\psi_h}}{Dr_h}, \ Dr_h \in \{Dr_h^c, Dr_h^w\}, \ \forall h \in \mathcal{H}, \ \forall n \in \mathcal{N} \tag{5}$$

Further, we define the computation capability of server $n \in \mathcal{N}$ as $f_{ser,n}$, then the execution time spent on server n is

$$d^{exe}_{\psi_h,n} = \frac{c_{\psi_h}}{f_{ser,n}}, \ \forall h \in \mathcal{H}, \ \forall n \in \mathcal{N} \tag{6}$$

Let $P_h \in \{P^c_h, P^w_h\}$ be the transmission power of UAV h via cellular or WiFi link, then given the transmission time $d^{tr}_{\psi_h,n}$, the amount of energy consumed by UAV h during data transmission can be calculated by

$$e^{tr}_{\psi_h,n} = P_h d^{tr}_{\psi_h,n}, \ \forall h \in \mathcal{H}, \ \forall n \in \mathcal{N} \tag{7}$$

The offloading nodes have the ability to efficiently transmit results, leading to a considerable decrease in downloading time compared to offloading time. As a result, it is acceptable to neglect the time duration and energy usage involved in downloading.

(3) U2X Offloading: In addition to establishing wireless connections and exchanging data with ground-based infrastructure like cellular BS or WiFi AP, UAVs also can communicate with the smart terminals such as other UAVs, vehicles, and IoT devices. When UAV $h \in \mathcal{H}$ offloads its task ψ_h to a smart terminal $m \in \mathcal{M}$ over an orthogonal channel, the data transmission rate Dr^m_h can be given by [41,42]

$$Dr^m_h = B_{h,m} \cdot \log_2(1 + \frac{P_{h,m} G_{h,m}}{(\sigma_{h,m})^2}) \tag{8}$$

where $B_{h,m}$ indicates the bandwidth of the wireless link between UAV h and terminal m, $P_{h,m}$ and $(\sigma_{h,m})^2$ represents the transmission power and additive white Gaussian noise power for offloading data from UAV h to terminal m. Further, the channel gain $G_{h,m} = \lambda_{h,m}|\chi_{h,m}|^2$, in which the path loss model for the large-scale fading gain is calculated as $\lambda_{h,m} = 133.06 + 31 \log_{10} l_{h,m}(dB)$, where $l_{h,m}$ represents the physical distance from UAV h to terminal m; and the random variable $\chi_{h,m}$ indicates a small-scale fading coefficient, $\chi_{h,m} \sim \mathcal{CN}(0,1)$.

Therefore, when task ψ_h is offloaded to the target smart terminal m, the transmission time is given by

$$d^{tr}_{\psi_h,m} = \frac{d_{\psi_h}}{Dr^m_h}, \ \forall h \in \mathcal{H}, \ \forall m \in \mathcal{M} \tag{9}$$

If UAV h offloads the computation task ψ_h to terminal m and successfully obtains the feedback associated with the computation results, denote $e^{tr}_{\psi_h,m}$ as the energy consumption of UAV h for uploading the task, which can be expressed by

$$e^{tr}_{\psi_h,m} = P_h d^{tr}_{\psi_h,m}, \ \forall h \in \mathcal{H}, \ \forall m \in \mathcal{M} \tag{10}$$

Further, let us define $f_{ter,m}$ as the computation capability of terminal m, the execution delay on terminal m for computing task ψ_h can be written as

$$d^{exe}_{\psi_h,m} = \frac{c_{\psi_h}}{f_{ter,m}}, \ \forall h \in \mathcal{H}, \ \forall m \in \mathcal{M} \tag{11}$$

Consequently, when UAV h performs its task ψ_h using any of the strategies considered in this paper, we define the total time experienced by the UAV h as $d^{\psi_h}_{total}$ for ease of expression. Specifically, $d^{\psi_h}_{total}$ can be given by

$$d^{\psi_h}_{total} = \begin{cases} d^{loc,exe}_{\psi_h}, & \text{if } a_h = 0, \text{ local computing} \\ d^{tr}_{\psi_h,n} + d^{exe}_{\psi_h,n}, & \text{if } a_h = 1, \text{ MEC offloading} \\ d^{tr}_{\psi_h,m} + d^{exe}_{\psi_h,m}, & \text{if } a_h = 2, \text{ U2X offloading} \end{cases} \tag{12}$$

In the system, prior to executing the offloading algorithm, UAVs can serve as data acquisition terminals for gathering sensor data within designated target regions. The energy consumed by the UAVs for hovering in these regions and executing data collection tasks precedes the offloading decisions made by the UAVs. However, the focus of this paper is not to investigate the energy consumed per se, but rather to explore the optimization of computational task offloading for UAVs, taking into account the energy consumption that directly impacts the offloading process. Consequently, UAVs have the option to offload computational tasks while continuing their predetermined operations. Similarly to the previous UAV offloading studies [25,36], during this stage, our attention is solely directed towards the energy consumed during the transmission process resulting from UAV offloading as described in Equations (7) and (10).

Thereby, when UAV h performs its task ψ_h using any of the involved strategies, our focus is on factors that affect UAV offloading, specifically the energy consumption of local computing on the UAV and the energy consumption of transmitting task data during offloading. In this case, the energy consumption for computation offloading denoted as $e_{total}^{\psi_h}$ can be calculated as

$$e_{total}^{\psi_h} = \begin{cases} e_{\psi_h}^{loc,exe}, & \text{if } a_h = 0, \text{ local computing} \\ e_{\psi_h,n}^{tr}, & \text{if } a_h = 1, \text{ MEC offloading} \\ e_{\psi_h,m}^{tr}, & \text{if } a_h = 2, \text{ U2X offloading} \end{cases} \tag{13}$$

As an additional extension, we have also taken into consideration the scenario in which the UAV hovers and waits for the offloading process to be completed, which results in hovering energy consumption. This hovering energy consumption has a significant impact on the optimization of the offloading process. In this case, the time that the UAV hovers in the region is considered to be the offloading time; let us denote the hovering time as $d_{\psi_h}^{hover}$, thereby, $d_{\psi_h}^{hover} \in \{d_{\psi_h,n}^{tr}, d_{\psi_h,m}^{tr}\}$. And, the hovering energy consumption is proportional to the hovering time $d_{\psi_h}^{hover}$, which is given by [43]

$$e_{\psi_h}^{hover} = P_{hover} d_{\psi_h}^{hover} = \frac{\nu\sqrt{\nu}}{p_e\sqrt{\frac{1}{2}\pi\omega_h {h_r}^2 D_0}} d_{\psi_h}^{hover} \tag{14}$$

in which $P_{hover} = \frac{\nu\sqrt{\nu}}{p_e\sqrt{\frac{1}{2}\pi\omega_h {h_r}^2 D_0}}$ represents the power required to keep the UAV hovering, taking into account the effects of factors such as air density D_0, power efficiency p_e, and the number ω_h and diameter h_r of the rotors. Additionally, ν denotes the total thrust generated by the UAV to counteract gravity and drag forces.

As a result, when considering the hovering energy consumption during offloading, using any of the computation offloading strategies involved, the energy consumption of the UAV can be calculated as

$$e_{total}^{\psi_h} = \begin{cases} e_{\psi_h}^{loc,exe}, & \text{if } a_h = 0, \text{ local computing} \\ e_{\psi_h,n}^{tr} + e_{\psi_h,n}^{hover}, & \text{if } a_h = 1, \text{ MEC offloading} \\ e_{\psi_h,m}^{tr} + e_{\psi_h,m}^{hover}, & \text{if } a_h = 2, \text{ U2X offloading} \end{cases} \tag{15}$$

3.3. Utility Function Design

For the applications executed, the utility of UAVs depends significantly on the utility of the attributes in terms of delay $d_{total}^{\psi_h}$ and energy consumption $e_{total}^{\psi_h}$. In order to effectively assess the delay and energy consumption generated by UAV $h \in \mathcal{H}$ when adopting different strategies, we utilize utility theory to design a utility function that maps these attributes to corresponding utility metrics, which will further account for the performance of the computation offloading. Considering that both delay and energy consumption are expected

to be as small as possible, a large delay will lead to poor performance, thus resulting in low utility, and it is also the same situation for the energy consumption. Therefore, we use z to denote the value of the generated delay $d_{total}^{\psi_h}$ or energy consumption $e_{total}^{\psi_h}$, i.e., $z \in \{d_{total}^{\psi_h}, e_{total}^{\psi_h}\}$.

According to the utility theory, the designed utility function is required to satisfy the properties of twice differentiable, monotonic, and concave-convex; in this case, the optimal point of the utility can be obtained. Moreover, when z starts to reduce from its maximum value within an allowable range, the utility function presents high sensitivity to the change in z , resulting in the utility rising significantly. Conversely, as the z gradually decreases towards the ideal value, typically zero, based on the marginal utility theory, the sensitivity of the utility to the change in the z diminishes, and the utility is becoming slower and slower. We define the maximum tolerable threshold of the attribute as z_{max}, considering the delay requirement d_{max} and energy limitation e_{max} of the UAV, here $z_{max} \in \{d_{max}, e_{max}\}$; further, the minimum value is z_{min} and $z_{mid} = \frac{(z_{min}+z_{max})}{2}$ is the intermediate value. The utility function denoted as $y(z)$ should satisfy three additional conditions:

$$y(z) = 1, \ \forall z \leq z_{min} \tag{16}$$

$$y(z) = 0, \ \forall z \geq z_{max} \tag{17}$$

$$y(z_{mid}) = 0.5, \ \forall z = z_{mid} \tag{18}$$

Let η_z ($\eta_z \geq 2$) signify the application's sensitivity to the attribute value, which has obvious impacts on the value of $y(z)$, then $y(z)$ can be formally given as

$$y(z) = \begin{cases} 1 & z = z_{min} \\ \frac{1}{1+(\frac{z}{z_{mid}-z_{min}})^{\eta_z}} & z_{min} < z \leq z_{mid} \\ \frac{(\frac{z_{max}-z}{z_{max}-z_{mid}})^{\eta_z}}{1+(\frac{z_{max}-z}{z_{max}-z_{mid}})^{\eta_z}} & z_{mid} < z < z_{max} \\ 0 & z \geq z_{max} \end{cases} \tag{19}$$

In the following, we will conduct a theoretical analysis and proof for the properties of the utility function illustrated in Equation (19).

Proof. Firstly, we can observe from Equation (19) that $y(z)$ satisfies the three constraints presented in Equations (16)–(18). Next, we demonstrate the three properties of the $y(z)$. The differentiability of the $y(z)$ can be proved through deriving the relationship between the left (right) derivative and the left (right) limit of the $y(z)$ on z_{mid}. Thus, we have

$$\lim_{z \to z_{mid}^{+}} y'(z) = \lim_{z \to z_{mid}^{+}} \frac{dy(z)}{dz} = -\frac{\eta_z}{4(z_{mid} - z_{min})} \tag{20}$$

$$\lim_{z \to z_{mid}^{-}} y'(z) = \lim_{z \to z_{mid}^{-}} \frac{dy(z)}{dz} = -\frac{\eta_z}{4(z_{max} - z_{mid})} \tag{21}$$

Since $z_{mid} = \frac{(z_{min}+z_{max})}{2}$, it can be deduced that

$$\lim_{z \to z_{mid}^{+}} y'(z) = \lim_{z \to z_{mid}^{-}} y'(z) = -\frac{\eta_z}{2(z_{max} - z_{min})} \tag{22}$$

Therefore, $y(z)$ is differentiable.

Subsequently, we validate the monotonicity and concavity-convexity for $y(z)$. When $z_{min} < z \leq z_{mid}$, the first and second derivatives of $y(z)$ can be represented, respectively, as follows:

$$y'(z) = \frac{dy(z)}{dz} = \frac{-\eta_z \cdot \left(\frac{z}{z_{mid}-z_{min}}\right)^{\eta_z - 1} \cdot \frac{1}{(z_{mid}-z_{min})}}{\left[1+\left(\frac{z}{z_{mid}-z_{min}}\right)^{\eta_z}\right]^2} = \frac{-\eta_z \cdot z^{\eta_z - 1}}{(z_{mid}-z_{min})^{\eta_z}\left[1+\left(\frac{z}{z_{mid}-z_{min}}\right)^{\eta_z}\right]^2} \tag{23}$$

$$y''(z) = \frac{dy'(z)}{dz} = \frac{-\eta_z \cdot z^{\eta_z - 2}\left[\eta_z - 1 - (\eta_z + 1)(\frac{z}{z_{mid}-z_{min}})^{\eta_z}\right]}{(z_{mid}-z_{min})^{\eta_z}\left[1+\left(\frac{z}{z_{mid}-z_{min}}\right)^{\eta_z}\right]^3} \tag{24}$$

In this case, $y'(z) \leq 0$ and $y''(z) \geq 0$. Meanwhile, when $z_{mid} < z < z_{max}$, $y'(z)$ and $y''(z)$ can be represented by Equations (25) and (26).

$$y'(z) = \frac{dy(z)}{dz} = \frac{-\eta_z \cdot (z_{max}-z)^{\eta_z - 1}}{(z_{max}-z_{mid})^{\eta_z}\left[1+\left(\frac{z_{max}-z}{z_{max}-z_{mid}}\right)^{\eta_z}\right]^2} \tag{25}$$

$$y''(z) = \frac{dy'(z)}{dz} = \frac{\eta_z \cdot (z_{max}-z)^{\eta_z - 2}\left[\eta_z - 1 - (\eta_z + 1)\left(\frac{z_{max}-z}{z_{max}-z_{mid}}\right)^{\eta_z}\right]}{(z_{max}-z_{mid})^{\eta_z}\left[1+\left(\frac{z_{max}-z}{z_{max}-z_{mid}}\right)^{\eta_z}\right]^3} \tag{26}$$

In this situation, $y'(z) < 0$ and $y''(z) < 0$ are still satisfied. Consequently, $y(z)$ will monotonically decrease with the variable z. In addition, $y(z)$ is concave when $z_{min} < z \leq z_{mid}$; conversely, when $z_{mid} < z \leq z_{max}$, $y(z)$ is convex.

The proof is accomplished. □

To this end, the delay and energy consumption utilities can be acquired according to Equation (19), which are represented as $y(d_{total}^{\psi_h})$ and $e(d_{total}^{\psi_h})$, respectively.

3.4. Problem Formulation

As stated above, no matter which strategy the UAV h adopts, based on multi-attribute utility theory, the utility of UAV h for processing task ψ_h can be expressed as the weighted utility accounting for $y(d_{total}^{\psi_h})$ and $y(e_{total}^{\psi_h})$, i.e.,

$$u_{h,a_h} = \delta_{\psi_h,d} \cdot y(d_{total}^{\psi_h}) + \delta_{\psi_h,e} \cdot y(e_{total}^{\psi_h}) \tag{27}$$

in which $\delta_{\psi_h,d}$ and $\delta_{\psi_h,e}$ signify the trade-off factors between delay and energy consumption.

Therefore, the objective of the UAV is to maximize its utility for processing a computation task through searching the optimal task offloading allocation strategy. Mathematically, the problem can be formulated as

$$\begin{aligned} &\max_{a_h} \sum_{h=1}^{H} u_{h,a_h} \\ =&\max_{a_h} \sum_{h=1}^{H} (\delta_{\psi_h,d} \cdot y(d_{total}^{\psi_h}) + \delta_{\psi_h,e} \cdot y(e_{total}^{\psi_h})) \\ s.t. \quad & C_1 : \sum a_h = 1,\ a_h \in \{0,1\} \\ & C_2 : C'_{ser,n}(t) \leq C_{ser,n},\ n \in \mathcal{N} \\ & C_3 : C'_{ter,m}(t) \leq C_{ter,m},\ m \in \mathcal{M} \\ & C_4 : \sum_{h=1}^{H} \alpha_{h,m}(t) \leq 1 \end{aligned} \tag{28}$$

where C_1 illustrates that the UAV h can select one of the available offloading nodes to process its task or choose to compute locally, and the problem is constrained by the binary offloading decisions. Further, C_2 and C_3 imply, respectively, that the computational

resources the offloading node $n \in \mathcal{N}$ and $m \in \mathcal{M}$ can provide at a certain time are limited by their computation capacity. C_4 indicates that the terminal $m \in \mathcal{M}$ is specified to provide computing services for, at most, one UAV within a certain time.

However, the UAV cannot be aware of the strategies of other UAVs; in addition, the servers' available computation capacity and the channel conditions are commonly unclosed to UAVs. As a result, the optimization problem should resort to a learning-based approach to solve.

3.5. A Preliminary Discussion on UAV Retransmission

Furthermore, if the UAV is processing a computation task that places stringent demands on the data transmission reliability, or is highly sensitive to precise decision making, a data retransmission check model can also be considered in the system to address this challenge and ensure data integrity and normal task execution. Due to space limitations, a preliminary and brief discussion of this topic will be presented in this paper.

Within the retransmission check model, the selected offloading node can send acknowledge character (ACK) messages to the UAV with service requests over a wireless link. These ACK messages serve as a feedback mechanism, allowing the UAV to ascertain the receiving status of the offloading target node. In particular, for maintaining the reliability and timeliness of the task data, and striking a balance between ensuring timely data reception and minimizing unnecessary retransmissions, each UAV imposes a delay constraint on the ACK messages and packages the information into time-stamped packets before uploading to its offloading target to verify whether the offloading target has successfully received the transmitted data within a predefined time window. If an offloading target fails to transmit the ACK message within the designated time window, it triggers the initiation of data retransmission. This mechanism enables the UAV to resend the unacknowledged data associated with the computation task, thereby guaranteeing the successful reception of the task data by the offloading target node. At this point, the time and energy consumed by the UAV retransmission will have an impact on the performance of UAV offloading.

4. Fuzzy Logic-Based Offloading Pre-Screening

Considering the movement of UAVs in system and the impact of transmission quality when offloading data, in this section, by leveraging the strengths of fuzzy logic theory on the adaptiveness to the varying environment and dealing with the uncertainty information through fuzzy inference, we devise a task offloading pre-screening mechanism that empowers UAVs to adaptively screen out the available offloading nodes from multiple servers and helper terminals.

As illustrated in Figure 2, the fuzzy logic-based pre-screening framework processes the UAV velocity relative to the potential offloading nodes and PLR generated during offloading with three procedures, i.e., fuzzification, fuzzy inference and defuzzification [44,45], and finally outputs a scalar value $Pr_s \in [0, 1]$. The Pr_s indicates the probability of selecting the offloading node s, representing adaptability degree of the task to node $s \in \mathcal{S}$, $\mathcal{S} = \mathcal{N} \cup \mathcal{M}$.

Particularly, the PLR generated in wireless transmission is evaluated by [46]

$$p^{tr}_{\psi_h,s} = \phi_1 \cdot d_{\psi_h} \cdot \exp(-\phi_2 \cdot \frac{P_{Rss}}{\sigma^2 \cdot l^2_{h,s}}), p^{tr}_{\psi_h,s} \leq p_{max} \tag{29}$$

where $0 < \phi_1, \phi_2 < 1$, both are the tunable parameters, P_{Rss} indicates the received signal strength, $\sigma^2 \in \{(\sigma^c)^2, (\sigma^w)^2, (\sigma^m)^2\}$ is the noise power, and $l_{h,s}$ denotes the distance from h to the node s.

Further, on the basis of the designed framework, a fuzzy logic-based offloading pre-screening scheme is devised to improve the transmission quality, and its detailed procedures are illustrated in Algorithm 1.

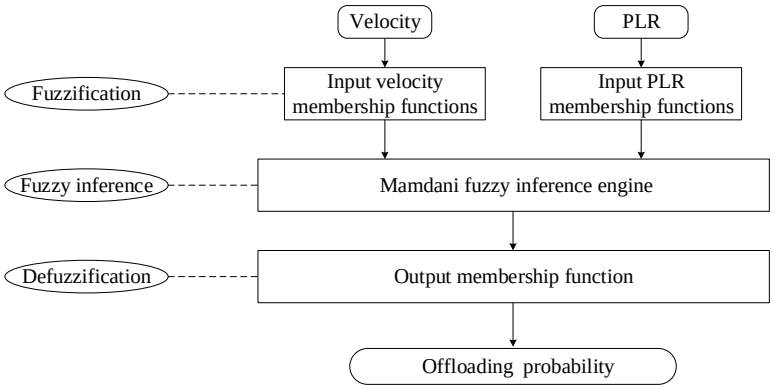


Figure 2. Fuzzy logic-based offloading pre-screening framework in DELOFF.

Algorithm 1 Fuzzy logic-based offloading pre-screening

Input: Potential offloading nodes set $\mathcal{S} = \mathcal{M} \cup \mathcal{N}$.

Output: Available offloading nodes set $\mathcal{S}$ for UAV h.

1: **while** each sense stage **do**
2: UAV $h \in \mathcal{H}$ senses existing offloading nodes;
3: **for** $s = 1 : S$ **do**
4: UAV h evaluates the velocity relative to node $s \in \mathcal{S}$;
5: UAV h measures the PLR $p^{tr}_{\psi_h,s}$ by using Equation (29);
6: $Pr_s \longleftarrow fuzzy(velocity\ and\ p^{tr}_{\psi_h,s})$;
7: **if** $Pr_s < \hat{P}r_s$ **then**
8: UAV h updates its available offloading node in $\mathcal{S}$
9: **end if**
10: **end for**
11: **end while**

The offloading pre-screening algorithm runs on each UAV in a decentralized manner, which takes the set $\mathcal{S} = \mathcal{M} \cup \mathcal{N}$ of all of the potential offloading nodes, including MEC servers and helper terminals, as the inputs for the purpose of previously evaluating and choosing available offloading nodes. In the algorithm proposed, in lines 2 to 5, each UAV h equipped with sensors periodically monitors the potential offloading nodes, samples the velocity relative to a specific offloading node, and measures the PLR in data transmission if assuming the task will be offloaded through the wireless link correspondingly. Then, in line 6, the algorithm proceeds to the developed fuzzy logic model. Particularly, in the fuzzification stage, the measured data in terms of velocity and PLR are first mapped to fuzzy sets based on the corresponding velocity and PLR, respectively. Afterward, the fuzzy inference operation analyzes the fuzzified inputs and generates the fuzzy output according to a series of fuzzy rules in the form of IF-AND-THEN that can express the relationship between the input variables and output. Furthermore, the defuzzification procedure acquires a scalar value Pr_s for each node $s \in \mathcal{S}$ by applying the centroid

defuzziffer method [47] under the triggered fuzzy rule. The $Pr_s \in [0,1]$ can be regarded as the adaptability degree of node s to task ψ_h, and a higher Pr_s means better adaptability. Thereby, finally, in lines 7 to 9, Pr_s is compared with a permitted threshold $\hat{Pr}_s$, and if the $Pr_s > \hat{Pr}_s$ holds, then the node is selected as the available offloading node for UAV h.

5. Decentralized Bandit Learning Solution for Task Offloading

In this section, we resort to the multi-arm bandit (MAB) learning to design a decentralized computation offloading framework for maximizing the long-term rewards of UAVs. Further, the procedures and learning regret of the algorithm are discussed, respectively.

5.1. Bandit Learning-Based Task Offloading Model

In particular, the optimization problem defined in Equation (28) is regarded as a MAB learning problem by modeling the UAVs as agents. For ease of illustration, we define the strategy space of the UAVs as $\mathcal{K} = \{1, 2, ..., K\}$; once the UAV choose one of the strategies, it is regarded as pulling an arm $k \in \mathcal{K}$, and the expected reward can be considered as $u_{h,k}$ according to u_{h,a_h} in Equation (28). Thereby, the problem in Equation (28) is transformed into the following:

$$\max \sum_{h \in \mathcal{N}_k} u_{h,k}\,, \quad s.t. \quad C_1 - C_4 \tag{30}$$

where $\mathcal{N}_k$ indicates the set of UAVs that choose to pull arm $k \in \mathcal{K}$.

Our formulated optimization problem belongs to a Generalized Assignment problem (GAP), it is commonly difficult for the algorithms with polynomial time complexity to find the optimal task offloading allocation strategies for UAVs. In addition, the suboptimal solution to our defined problem resulted from a centralized approach and only guarantees that $(1+\beta)$-approximate utility, i.e, $\sum_{h=1}^{\mathcal{H}} u_{h,a_h} \geq \frac{1}{1+\beta} \sum_{h=1}^{\mathcal{H}} u_{h,a_h^*}$, in which β is the approximate ratio, $\beta \geq 1$, and a_h^* is the oracle strategy to the GAP problem. As a result, in the MAB model, let r_{h,a_h} be the reward perceived by the UAV for making an offloading strategy at time slot t in a decentralized manner, and $\mathbb{E}[r_{h,a_h}(t)] = u_{h,a_h}$ when the computation task is accomplished, otherwise $r_{h,a_h} = 0$. Furthermore, by utilising the highest suboptimal utility $\frac{T}{2} \sum_{h=1}^{\mathcal{H}} u_{h,a_h^*}$, i.e., $\beta = 1$, for the bandit learning based task offloading model, the learning regret can be represented as

$$\mathcal{R}(T) = \frac{T}{2} \sum_{h=1}^{\mathcal{H}} u_{h,a_h^*} - \sum_{t=1}^{T} \sum_{h=1}^{\mathcal{H}} \mathbb{E}[r_{h,a_h}(t)] \tag{31}$$

To achieve desirable service performance or minimize the learning regret $\mathcal{R}(T)$, the trade-off between the exploration and exploitation should be effectively considered in the design of the DELOFF scheme. Thereby, in DELOFF scheme, we divide the entire time domain $\mathcal{T}$ into an epoch sequence $\{1, ..., I_T\}$ during the learning process, where each epoch consists of a varying number of time slots, and the index I_T represents the final epoch in the epoch sequence. Moreover, to facilitate the trade-off between exploration and exploitation, the learning process within each epoch is decomposed into three stages: exploration, assignment, and exploitation. Specifically, in the i-th epoch, the exploration procedure occupies the initial T_1 time slots, and the assignment stage requires the subsequent T_2 time slots, where $T_1 = K$, and T_2 are at most equal to S. Finally, the exploitation stage extends for the remaining time slots within the epoch, and the number of it is 2^i. This means the exploitation dominates the exploration and assignment procedures; thus, the exponential time slots are taken.

(1) Exploration: In this stage, each UAV $h \in \mathcal{H}$ traverses the available offloading nodes in $\mathcal{S}$, screened by performing the proposed offloading pre-screening mechanism and offloading its task, so as to obtain the reward $\tilde{r}_{h,k}^{(i)}$, which represents an empirical (sample-mean) estimation of the reward r_{h,a_h} for pulling arm k from the beginning to the current time slot during the learning process. In more detail, if arm k has been selected

$N_t(k)$ times up until the end of time T, the empirical mean reward for pulling arm k can be represented as $\frac{\sum_{j=1}^{N_t(k)} r_{h,a_h}[j]}{T}$, where $\sum_{j=1}^{N_t(k)} r_{h,a_h}[j]$ indicates the sum of the reward r_{h,a_h} for selecting arm k with $N_t(k)$ times.

(2) Assignment: This stage mainly aims to achieve the $(1+\beta)$-approximate task offloading assignment for the problem defined in Equation (28). To achieve a 2-approximate assignment, we initially decouple the assignment problem into S knapsack sub-problems, and subsequently obtain the optimal solution for each sub-problem by using the branch-and-bound (BnB) approach as the $(1+\beta)$ approximation oracle to create matching between the tasks of UAVs and offloading node $s \in \mathcal{S}$ sequentially. Specifically, we use an indicator function $\boldsymbol{\xi} = \{\xi_h, h \in \mathcal{H}\}$ to store the current assignment results of the computation task for each UAV h. For the ξ_h, $\xi_h = 0$ means, the UAV h chooses to computing its task locally, and $\xi_h = k'$ implies that UAV h will change its matching to k'. During this stage, we define a function as $\Delta\tilde{r}_{h,k}^{(i)}$ to characterize the performance improvement of the computation task for the UAV, i.e.,

$$\Delta\tilde{r}_{h,k}^{(i)} = \begin{cases} \tilde{r}_{h,k}^{(i)} & \xi_h = 0 \\ \tilde{r}_{h,k'}^{(i)} - \tilde{r}_{h,k}^{(i)} & \xi_h = k' \end{cases} \tag{32}$$

Accordingly, given the performance improvement $\Delta\tilde{r}_{h,k'}^{(i)}$, through continuously updating the assignment ξ_h based on the BnB, the final assignment $\boldsymbol{\xi}$ can achieve a 2-approximate when all of the sub-problems can be processed optimally.

(3) Exploitation: Exploitation enables the agent to focus on arms that have historically provided higher rewards. Therefore, this stage dominates the exploration and assignment procedures, thus the exponential time slots are taken, and all UAVs will offload their computation tasks by exploiting the assignment $\boldsymbol{\xi}$ to maximize the overall cumulative rewards over time.

5.2. The Proposed DELOFF Algorithm

The details of the proposed decentralized offloading scheme is summarized in Algorithm 2.

In the MAB-based DELOFF algorithm, each UAV initially filters the available offloading nodes using Algorithm 1, based on the quality of wireless links transmission to reduce the strategy space of the learning scheme. Then, the algorithm proceeds to the bandit learning procedures. During the learning process, the algorithm enables each UAV h to keep track of the empirical mean reward $\tilde{r}_{h,k}^{(i)}$ for pulling each arm $k \in \mathcal{K}$. Furthermore, as outlined in Section 5.1, the DELOFF algorithm consists of three different stages, which are exploration (lines 6∼11), task offloading assignment (lines 12∼20), and exploitation (lines 21∼24). In the exploration stage, all UAVs will pull the arm $k \in \mathcal{K}$ at least once. Afterward, each UAV m observes the reward $r_{h,a_h}(t)$ and updates $\tilde{r}_{h,k}^{(i)}$ with $r_{h,a_h}(t)$ obtained for selecting arm k. Next, the algorithm proceeds to the assignment stage, where all UAVs locally calculate the performance improvement of the reward using Equation (32), i.e., $\Delta\tilde{r}_{h,k}^{(i)}$, and send $\Delta\tilde{r}_{h,k}^{(i)}$ to the corresponding offloading node $s \in \mathcal{S}$ to execute the BnB algorithm. Further, if the task of UAV h is allocated to the offloading node s to process, we need to update ξ_h. When the exploitation stage starts, all UAVs exploit the assignment result $\boldsymbol{\xi}$ to process their tasks for 2^i time slots. Finally, the optimal task allocation $\mathbf{a} = \{a_h, h \in \mathcal{H}\}$ for computation offloading can be derived if the learned $\tilde{r}_{h,k}^{(i)}$ approximately approaches to the expected reward u_{h,a_h}, corresponding to the arm k.

The complexity of Algorithm 1 is $O(S)$, which results from the loop of scanning the potential offloading nodes. And, the complexity of Algorithm 2 is $O(HS)$ due to the complexity of implementing assignments between multi-UAVs and their filtered available nodes, while the reward update procedures only take constant time. Hence, the complexity of DELOFF is $O(HS)$ in the worst case.

Algorithm 2 MAB-based Decentralized Offloading Algorithm

Input: $\{C_{ser,n}, n \in \mathcal{N}\}$, $\{C_{ter,m}, m \in \mathcal{M}\}$, $\{c_{\psi_h}, h \in \mathcal{H}\}$

Output: the optimal allocation for computation offloading $\mathbf{a} = \{a_h, h \in \mathcal{H}\}$

Initialize $\mathcal{S} = \mathcal{N} \cup \mathcal{M}$, $\{\tilde{r}_{h,k}^{(i)} = 0, \forall h \in \mathcal{H}, k \in \mathcal{K}\}$;
for each UAV $h \in \mathcal{H}$ **do**
  Obtain all the available offloading nodes $\mathcal{S}$ by executing Algorithm 1;
end for
for epoch $i = 1 : I_T$ **do**
  #Exploration stage
  Set $T_1 \leftarrow K$
  for $t = 1 : T_1$ **do**
    For each UAV h, traverses all the available strategies in $\mathcal{K}$ at least once;
    Update $\tilde{r}_{h,k}^{(i)}$ by $r_{h,a_h}(t)$ obtained for playing arm k;
  end for
  #Assignment stage
  Set the assignment indicator $\boldsymbol{\xi} = \{\xi_h = 0 \, h \in \mathcal{H}\}$;
  Set $T_2 \leftarrow S$;
  for $s = 1 : S$ **do**
    Set $k \leftarrow s$;
    Each UAV calculates $\Delta\tilde{r}_{h,k}^{(i)}$ using Equation (32);
    Execute BnB for node $s \in \mathcal{S}$ with the inputs $\{C_{ser,n}, n \in \mathcal{N}\}$, $\{C_{ter,m}, m \in \mathcal{M}\}$, $\{c_{\psi_h}, h \in \mathcal{H}\}$, and $\{\Delta\tilde{r}_{h,k}^{(i)}, h \in \mathcal{H}\}$;
    Each UAV h updates ξ_h if its task is assigned to the node s;
  end for
  #Exploitation stage
  for the rest 2^i time slots **do**
    Each UAV h offloads task by following the assignment results $\boldsymbol{\xi}$ and update $\tilde{r}_{h,k}^{(i)}$ by $r_{h,a_h}(t)$.
  end for
end for

5.3. Regret Analysis

The upper bound of learning regret is demonstrated in this subsection, so as to measure performance loss, and show robustness of the proposed MAB-based decentralized offloading scheme.

According to Equation (32), we denote two types of the instant reward gap between two UAVs h and h' i.e., $g_{min}^{(1)}$ and $g_{min}^{(1)}$, which can be given by

$$g_{min}^{(1)} = \min_{h,h' \in \mathcal{H}} \min_{j,k \in \mathcal{K}} |u_{h,j} - (u_{h',j} - u_{h',k})|, \; h \neq h' \in \mathcal{H} \tag{33}$$

$$g_{min}^{(2)} = \min_{h,h' \in \mathcal{H}} \min_{j,j',k \in \mathcal{K}} |(u_{h,j} - u_{h,j'}) - (u_{h',j} - u_{h',k})|, \; h \neq h \in \mathcal{H}, \; j \neq j' \neq k \in \mathcal{K} \tag{34}$$

We then define the minimum reward gap of UAV h for pulling arms k and k' as g_{min}, i.e., $g_{min}^{(3)} = \min_{h \in \mathcal{H}} \min_{k,k' \in \mathcal{K}} \{|u_{h,k} - u_{h,k'}|\}$. Further, a reward gap denoted by g_{min} is given by

$$g_{min} = \min\{g_{min}^{(1)}, g_{min}^{(2)}, g_{min}^{(3)}\} \tag{35}$$

In addition, the ratio between the maximum capacity $C_s = \max\{C_{ser,n}, C_{ter,m}, \forall n \in \mathcal{N}, m \in \mathcal{M}\}$ and the minimum computation requirement of the tasks can be represented as $H_{max} = \frac{C_s}{\min_{h \in \mathcal{H}}\{c_{\psi_h}\}}$, which also indicates the maximum number of UAVs for which the system can provide the computing service at a certain time slot.

Next, on the basis of the above definitions, Theorem 1 reveals that the learning regret presented in Equation (31) has a tight upper bound.

Theorem 1. *Let us denote* $T_1 \approx \lceil K \frac{25 H_{max}^2 \Delta_r^2}{2(g_{min})^2} \rceil$ *when the number of UAVs is less than that of their available offloading nodes, i.e.,* $H < K$*; otherwise,* $T_1 \approx \lceil (K+H) \frac{25 H_{max}^2 \Delta_r^2}{2(g_{min})^2} \rceil$*, where* $\Delta_r = \bar{r} - \underline{r}$ *indicates the difference between the upper and lower bound of the reward obtained by the UAV in system. The upper bound of the regret in DELOFF can be expressed by*

$$\tilde{\mathcal{R}}(\mathcal{T}) \leq (T_1 \frac{H\bar{r}}{2} + T_2 \frac{H\bar{r}}{2}) \cdot \log_2(T+2) + 4H^2 K \bar{r} = O(\log_2 T) \tag{36}$$

Proof. Firstly, in the exploration stage, we establish an upper limit on the exploration error probability, i.e., $\epsilon = \{|\tilde{r}_{h,k}^{(i)} - u_{h,k}| > \frac{g_{min}}{5H_{max}}\}$. After conducting the i-th exploration, any UAV $h \in \mathcal{H}$ would perceive reward for pulling arm $k \in \mathcal{K}$ by at least ρ_{min} times, here $\rho_{min} = \frac{T_1}{K} \cdot i = \lceil \frac{25 H_{max}^2 \Delta_r^2}{2(g_{min})^2} \rceil \cdot i$. Thereby, the probability ϵ follows

$$\begin{aligned} \mathbf{Pr}(\epsilon | \rho_{min}) &\leq \bigcup_{h=1}^{H} \bigcup_{k=1}^{K} \mathbf{Pr}(|\tilde{r}_{h,k}^{(i)} - u_{h,k}| > \frac{g_{min}}{5H_{max}}) \\ &\leq H \cdot K \cdot \max_{h \in \mathcal{H},\, k \in \mathcal{K}} \mathbf{Pr}(|\tilde{r}_{h,k}^{(i)} - u_{h,k}| > \frac{g_{min}}{5H_{max}}) \end{aligned} \tag{37}$$

According to Chernoff–Hoeffding inequality, Equation (37) satisfies that

$$\mathbf{Pr}(\epsilon | \rho_{min}) \leq 2H \cdot K \cdot e^{-\frac{2(g_{min})^2}{25 H_{max}^2 \Delta_r^2} \rho_{min}} \leq 2H \cdot K \cdot e^{-i} \tag{38}$$

In the assignment stage, according to Equation (38), it has $|\tilde{r}_{h,k}^{(i)} - u_{h,k}| \leq \frac{g_{min}}{5H_{max}}$ at least with a probability of $1 - 2H \cdot K \cdot e^{-i}$. Let $\boldsymbol{\xi}^{(u)} = \{\xi_h^{(u)}, h \in \mathcal{H}\}$ be the indicator that determines the matching strategy $\mathbf{a}^{(u)}$ for the optimization problem described in Equation (28) based on the expected rewards $\{u_{h,k}\}$. For the subproblem described in

Equation (30) that corresponds to the offloading node $k \in \mathcal{K}$ under optimal matching, we denote

$$opt_k^1(\mathbf{a}^{(u)}) = \sum_{h=1}^{H} \Delta u_{h,k} = \sum_{h=1}^{H} (u_{h,k} - u_{h,\xi_h^{(u)}}) \tag{39}$$

$$opt_k^2(\mathbf{a}^{(u)}) = \sum_{h=1}^{H} \Delta \tilde{r}_{h,k}^{(i)} = \sum_{h=1}^{H} (\tilde{r}_{h,k}^{(i)} - \tilde{r}_{h,\xi_h^{(u)}}^{(i)}) \tag{40}$$

Then, it can be deduced that

$$opt_k^1(\mathbf{a}^{(u)}) - opt_k^2(\mathbf{a}^{(u)}) = \sum_{h=1}^{H} [(u_{h,k} - \tilde{r}_{h,k}^{(i)}) - (u_{h,\xi_h^{(u)}} - \tilde{r}_{h,\xi_h^{(u)}}^{(i)})] \tag{41}$$

Let $\zeta_{h,k} = (\tilde{r}_{h,k}^{(i)} - u_{h,k})$, considering that each offloading node can host a maximum of H_{max} UAVs, and $|\tilde{r}_{h,k}^{(i)} - u_{h,k}| \leq \frac{g_{min}}{5H_{max}}$; thereby, we have

$$opt_k^1(\mathbf{a}^{(u)}) - opt_k^2(\mathbf{a}^{(u)}) \leq H_{max} \cdot \max\{|\zeta_{h,k}| + |\zeta_{h,\xi_h^{(u)}}|\} \leq \frac{2g_{min}}{5} \tag{42}$$

Further, given the optimal matching $\mathbf{a}^*$, as similar to Equations (39) and (40), we have

$$opt_k^1(\mathbf{a}^*) = \sum_{h=1}^{H} \Delta u_{h,k} = \sum_{h=1}^{H} (u_{h,k} - u_{h,\xi_h}) \tag{43}$$

$$opt_k^2(\mathbf{a}^*) = \sum_{h=1}^{H} \Delta \tilde{r}_{h,k}^{(i)} = \sum_{h=1}^{H} (\tilde{r}_{h,k}^{(i)} - \tilde{r}_{h,\xi_h}^{(i)}) \tag{44}$$

Hence, similarly to Equation (42), it can be deduced that

$$opt_k^1(\mathbf{a}^*) - opt_k^2(\mathbf{a}^*) \leq H_{max} \cdot \max\{|\zeta_{h,k}| + |\zeta_{h,\xi_h}|\} \leq \frac{2g_{min}}{5} \tag{45}$$

Moreover, $opt_k^2(\mathbf{a}^{(u)}) \leq opt_k^2(\mathbf{a}^*)$ due to $\mathbf{a}^*$ is the optimal matching for the offloading node $k \in \mathcal{K}$ given the rewards $\{\Delta \tilde{r}_{h,k}^{(i)}, h \in \mathcal{H}\}$. In this case, the following inequality can be derived,

$$\begin{aligned} & opt_k^1(\mathbf{a}^{(u)}) - opt_k^1(\mathbf{a}^*) \\ & = opt_k^1(\mathbf{a}^{(u)}) - opt_k^2(\mathbf{a}^{(u)}) + opt_k^2(\mathbf{a}^{(u)}) - opt_k^1(\mathbf{a}^*) \\ & \leq [opt_k^1(\mathbf{a}^{(u)}) - opt_k^2(\mathbf{a}^{(u)})] + [opt_k^2(\mathbf{a}^*) - opt_k^1(\mathbf{a}^*)] \\ & \leq \frac{4g_{min}}{5} \end{aligned} \tag{46}$$

Moreover, $opt_k^1(\mathbf{a}^{(u)}) - opt_k^1(\mathbf{a}) \geq g_{min}, \exists k \in \mathcal{K}$. If $opt_k^1(\mathbf{a}^{(u)}) - opt_k^1(\mathbf{a}^*) \leq \frac{4g_{min}}{5}, \forall k \in \mathcal{K}$, it is highly likely that $opt_k^1(\mathbf{a}^{(u)}) = opt_k^1(\mathbf{a}^*)$ for each $k \in \mathcal{K}$, i.e., $\boldsymbol{\xi} = \boldsymbol{\xi}^{(u)}$. Consequently, we can assure that $\mathbf{a}^*$ is the 2-approximate strategy for our optimization problem in Equation (28).

Finally, the exploitation stage lasts for 2^i time slots and exploits the optimal indicator $\boldsymbol{\xi}^{(u)}$ to maximize the overall cumulative rewards over time.

For the existence of the exploration, assignment, and exploitation stages in the proposed DELOFF, the cumulative regret $\tilde{\mathcal{R}}(T)$ in Equation (31) can mainly be determined by the time slots corresponding to different stages, the inequalities given in Equation (38), the number of UAVs, and the upper bound reward $\bar{r}$ of the UAV. To sum up, $\tilde{\mathcal{R}}(T)$ satisfies the following expression:

$$\begin{aligned}
\tilde{\mathcal{R}}(T) &\leq \sum_{i=1}^{I_T}(T_1\frac{H\bar{r}}{2} + T_2\frac{H\bar{r}}{2} + 2HKe^{-i}\cdot T_3\cdot\frac{H\bar{r}}{2}) \\
&\leq \sum_{i=1}^{I_T}(T_1\frac{H\bar{r}}{2} + T_2\frac{H\bar{r}}{2} + 2HKe^{-i}\cdot 2^i\cdot\frac{H\bar{r}}{2}) \\
&= \sum_{i=1}^{I_T}(T_1\frac{H\bar{r}}{2} + T_2\frac{H\bar{r}}{2}) + H^2K\bar{r}\cdot\sum_{i=1}^{I_T}(\frac{2}{e})^i \\
&\leq (T_1\frac{H\bar{r}}{2} + T_2\frac{H\bar{r}}{2})I_T + H^2K\bar{r}\cdot\frac{2}{e-2} \\
&\leq (T_1\frac{H\bar{r}}{2} + T_2\frac{H\bar{r}}{2})\log_2(T+2) + H^2K\bar{r}\cdot\frac{2}{e-2} \\
&\leq (T_1\frac{H\bar{r}}{2} + T_2\frac{H\bar{r}}{2})\log_2(T+2) + 4H^2K\bar{r} \\
&= O(\log_2\mathcal{T})
\end{aligned} \tag{47}$$

which accomplishes the proof. □

6. Simulation Results

In this section, we evaluate the performance of the DELOFF mechanism from multiple aspects. The task offloading scenario is covered by heterogeneous networks over an 800 m × 800 m × 800 m area, where multi-UAVs are randomly distributed. The communication radius of cellular and WiFi networks are 400 m and 200 m, respectively, and the U2X communication radius is 30 m [48–50]. The bandwidths B^c and B^w for cellular and WiFi access are set to 10 MHz and 20 MHz, respectively, as well as the bandwidth $B_{m,h}$, which is 5 MHz [51–53]. The transmission power P_h^c, P_h^w, and $P_{h,m}$ are set to be the same value, i.e., 15 W, and the noise $(\sigma^c)^2$, $(\sigma^w)^2$ and $(\sigma_{h,m})^2$ are −89 dBm [54]. Moreover, the computation capacity of server $n \in \mathcal{N}$, helper terminal $m \in \mathcal{M}$, and UAV $h \in \mathcal{H}$, denoted by $f_{ser,n}$, $f_{ter,m}$, and $f_{uav,h}$, follow the uniform distribution in $[8, 10]$ G cycles/s, $[3, 4]$ G cycles/s, and $[1, 2]$ G cycles/s, respectively. Meanwhile, the edge servers and helper terminals endow maximum computation capacity denoted as $C_{ser,n}$, $C_{ter,m}$, which are uniformly distributed in $[18, 20]$ and $[2.5, 3]$ G cycles. In addition to the performance comparison performed in the scenario of varying task data size in Section 6.5, in other simulations for generated computation tasks, the data size d_{ψ_h} is randomly distributed in $[2.8, 3]$ MB. Similarly, in scenarios that do not consider the impact of changing computation resource demands, the required computation resources c_{ψ_h} are set to be randomly distributed in $[2, 2.5]$ G cycles. Note that these values represent a typical range of data and computational workloads commonly encountered in UAV tasks. The actual data size depends on factors such as the data format, while the computational workload depends on the complexity of the algorithms applied. To create a balance between delay and energy consumption when evaluating the offloading performance, both $\delta_{\psi_h,d}$ and $\delta_{\psi_h,e}$ are set to be 0.5 accordingly. Finally, Table 2 lists all of the key parameters used in the simulations.

Table 2. Parameter Settings.

Symbol	Value	Symbol	Value
B^c (MHz)	10	B^w (MHz)	20
$B_{m,h}$(MHz)	5	$(\sigma^c)^2, (\sigma^c)^w, (\sigma_{h,m})^2$ (dBm)	−89
P_h^c, P_h^w, $P_{h,m}$(W)	15	$f_{uav,h}$(G cycles/s)	$[1, 2]$
$f_{ter,m}$(G cycles/s)	$[3, 4]$	$f_{ser,n}$ (G cycles/s)	$[8, 10]$
$C_{ter,m}$(G cycles)	$[2.5, 3]$	$C_{ser,n}$ (G cycles)	$[18, 20]$

6.1. Probability Analysis of Offloading Nodes

We first present the offloading probability to any of the potential offloading nodes for the UAV under different relative velocities and PLRs in Figure 3. We discover that the probability declines with the increment of the velocity and PLR as expected; this situation demonstrates that our designed offloading pre-screening scheme is correct and effective for assisting and screening the potential offloading nodes in the system.

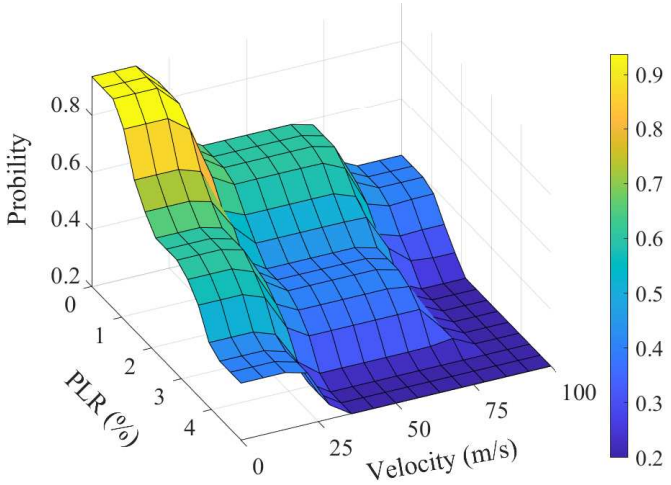


Figure 3. Offloading probability under different velocities and PLRs.

6.2. Convergence Evaluation

Figure 4 shows the time accumulated reward of DELOFF proposed over the iteration times when varying the number of UAVs. It can be seen that the curves of all accumulated rewards present an upward trend with the increase in iterations. This situation demonstrates the effectiveness of the DELOFF scheme. Furthermore, as shown in Figure 5, we introduce the DEL-WF as a baseline and make a convergence comparison when varying the number of UAVs; the DEL-WF scheme is originated from the DELOFF but without considering the fuzzy logic-based pre-screening mechanism proposed in DELOFF. It can be observed that the reward of DELOFF can stabilize over the time iterations rapidly, and the convergence performance of the DELOFF proposed is better than that of the DEL-WF as expected. This validates that the DELOFF can eliminate many offloading nodes according to service performance of the potential nodes to search for the optimal solution, which also promotes convergence effectively.

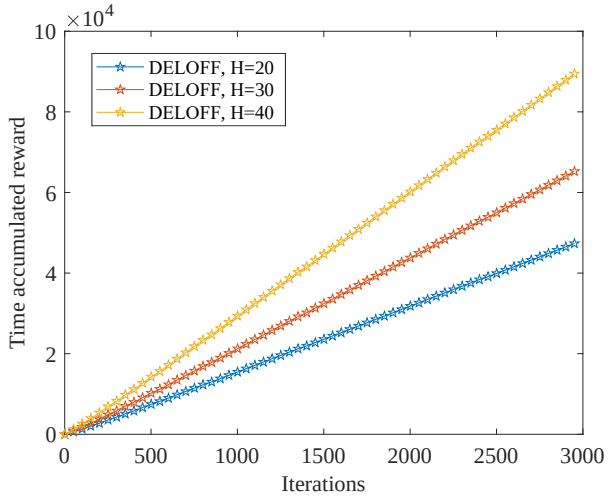


Figure 4. Time accumulated reward when varying number of UAVs.

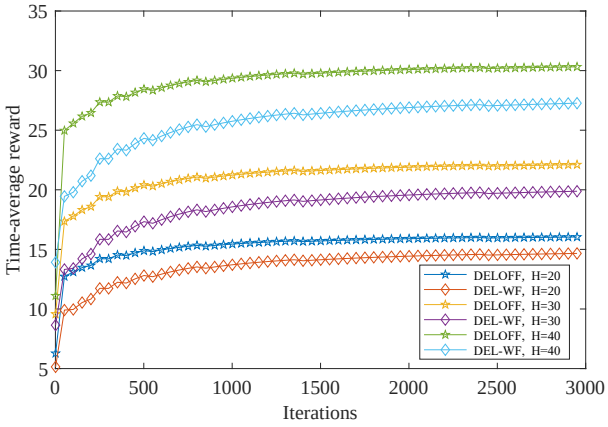


Figure 5. Convergence comparison on time-average reward.

6.3. Performance Comparison with Varying Number of Tasks

We carry out a series of performance comparisons between the DELOFF proposed and existing mainstream offloading schemes. For comparison, four comparative algorithms are considered including (1) An upper confidence bound-based task offloading scheme (UCBTO) [55]; (2) A weight improvement-based particle swarm optimization offloading algorithm (IWPSO) [56]; (3) A task offloading based on an improved genetic algorithm (IGATO) [57]; (4) A greedy-based sequential tuning computation offloading scheme (STCO) [58].

As depicted in Figures 6–10, we investigate the achieved performance of different offloading schemes in terms of different performance metrics for UAVs during offloading tasks, where the delay and energy consumption of each UAV to complete the computation offloading are calculated accordingly using Equations (12) and (15). In Figure 6, it is observed that the delay realized in different approaches raises with the increment of the tasks. This is because the increase in tasks at a certain time causes high computation demands and high delay. However, this upward trend is moderate for the DELOFF proposed in contrast to other schemes. And, the delay is reduced up to 12.05% over the IWPSO algorithm, which demonstrates that the DELOFF scheme can adaptively achieve the optimal computation offloading when considering U2X offloading and MEC offloading. On the other side, the IGATO wants to minimize its overall task offloading cost rather than taking the cost minimization into account for individual UAVs with computation tasks, and the average delay performance consequently performs more poorly than others.

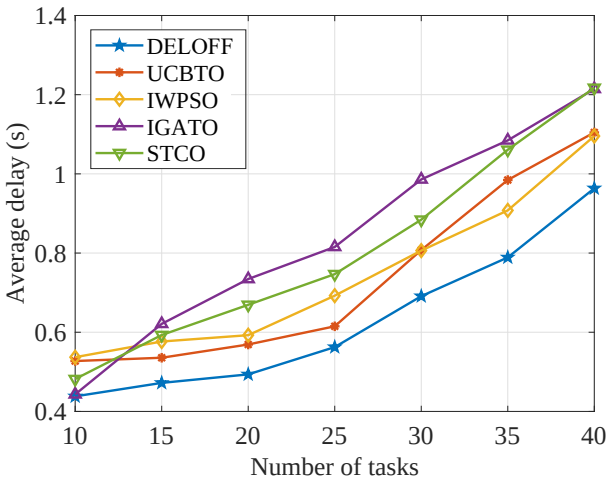


Figure 6. Average delay versus the number of tasks.

From Figure 7, we can find that the average energy consumed by each UAV in five algorithms increases with the increment of number of tasks. This is because with the increase in tasks processed, the UAVs need more energy to process the data in local, or offload them to the helper terminals or MEC servers according to the designed strategies. However, the consumed energy of the proposed DELOFF scheme rises gradually and is lower than that of the other comparative methods by at least 17.09%. It occurs because our mechanism explicitly and effectively minimizes the time of processing the tasks through adopting appropriate offloading strategies, and facilitates the reduction in the energy consumption of UAVs. Moreover, we observe that the performance achieved by the UCBTO algorithm and the IWPSO algorithm in Figures 6 and 7 is relatively similar when the number of tasks is small. However, as the number of tasks increases, the IWPSO algorithm shows potential for relatively superior performance. This can be attributed to the advantage of IWPSO in the swarm-based optimization capability. As the number of tasks increases, the IWPSO algorithm has more opportunities to efficiently explore various solutions and discover a more optimal strategy that minimizes delay and energy consumption. Compared to the UCBTO algorithm, the IWPSO algorithm achieves relatively improved task allocation and resource utilization in scenarios with a higher number of tasks, effectively reducing delay and energy consumption. Consequently, the performance trends of IWPSO exhibit a relatively gradual change as the number of tasks increases.

In general, our DELOFF proposed performs better than other methods by distributing the tasks of UAVs to the available offloading nodes through bandit feedbacks.

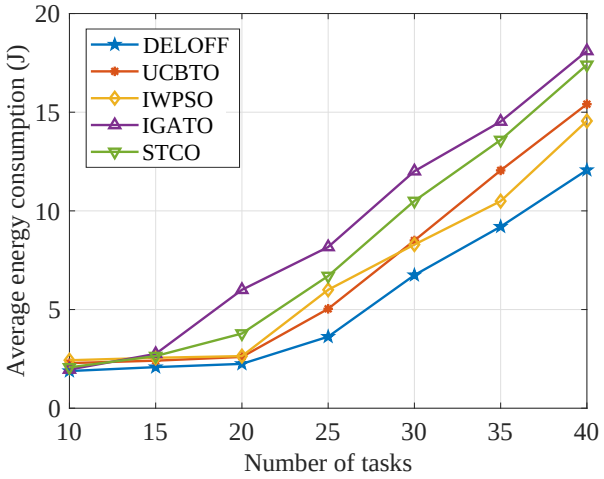


Figure 7. Average energy consumption versus the number of tasks.

Additionally, the overall PLR that occurred during the data transmission process is measured in Figure 8. It reflects that the DELOFF proposed outperforms the other four task offloading approaches in terms of PLR performance. This is because our devised task offloading model can always make the most effective strategy through fully considering the impact of PLR on the integrity of the transmitted data compared with other schemes. The numerical results indicate that the total PLR during offloading in our DELOFF scheme is lower than that in other algorithms by at least 12.99%.

In the model considered, when the DELOFF scheme violates at least one constraint, we treat it as task failure. Figure 9 shows the overall task failures under different numbers of tasks during the task offloading process; we observe that our mechanism can always keep task failure at a lower level. The DELOFF takes advantage of the fuzzy logic-based policy on conducting effective filtering to offloading nodes by deriving a self-adapting probability of accessing the link according to the PLR and velocity, which is helpful for avoiding disconnection or poor transmission quality. On this basis, our proposed DELOFF attentively offloads the tasks by jointly considering the delay and energy, and can satisfy the multiple constraints of a higher number of UAVs with computation tasks to process.

In contrast, the other schemes neglect the relationship between task failure and other performance indicators, leading to more task failures as compared to the proposed DELOFF. In summary, the DELOFF can reduce the number of task failures by at least 15.83%.

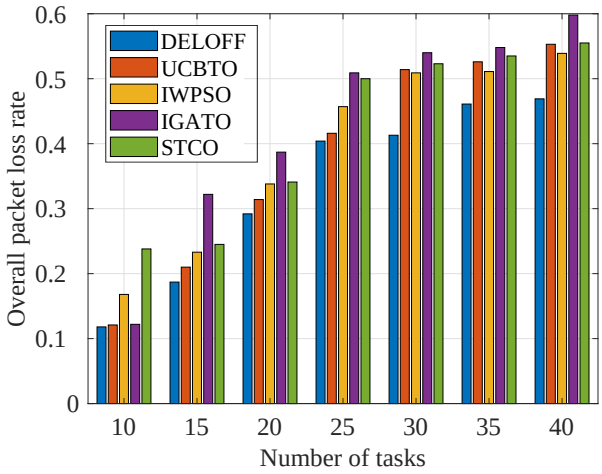


Figure 8. Packet loss rate versus the number of tasks.

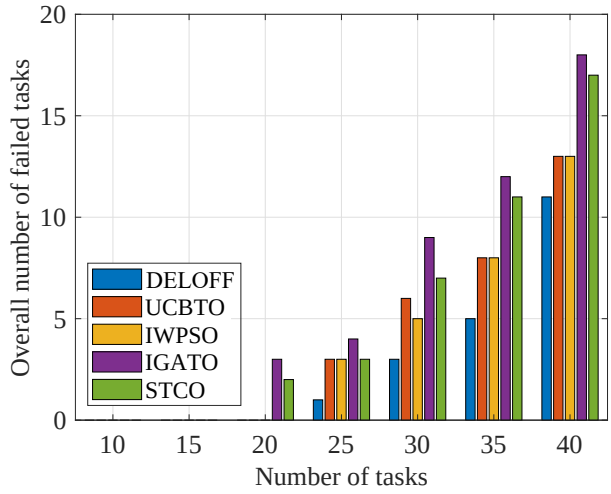


Figure 9. Task failures versus the number of tasks.

Figure 10 displays the average utility with respect to the different number of tasks. As can be seen, the average utilities in different schemes are almost close to each other when the number of tasks processed is small, while the DELOFF scheme is able to achieve the highest utilities for UAVs in the four benchmark schemes. This is because the designed mechanism of DELOFF can derive optimal task offloading strategies by identifying the available offloading nodes as well as benefiting from the bandit learning with implicit feedback. Accordingly, more tasks of UAVs are facilitated to be effectively processed to obtain higher utility under the capacity constraints of servers. As a summary, the DELOFF improves up to 11.82% utility over the four schemes, respectively.

In addition, we preliminarily explore the impact of retransmissions on the offloading performance of the DELOFF proposed. Specifically, we investigate the influence of increasing the maximum allowed number of retransmissions on the delay and energy consumption of the UAV offloading as the number of computational tasks increases, following the similar approach described in [59]. Based on the results shown in Figure 11a,b, it is evident that increasing the maximum allowed number of retransmissions leads to elevated data transmission latency and increased energy consumption. In particular, excessive retransmissions can cause significant delays in data transmission, being especially worse for time-sensitive tasks. Additionally, even with multiple retransmissions, there

is still a possibility of unsuccessful transmission if the distance between the offloading node and the UAV exceeds a certain communication range. Consequently, unnecessary retransmissions can negatively affect the performance of UAV offloading and the overall operational efficiency of the UAV network. Therefore, in order to adaptively adjust the retransmission strategy for computation task data, we can further consider the sensitivity of different computational tasks to data loss and the impact of retransmissions on offloading performance. This issue will be addressed as part of our future work.

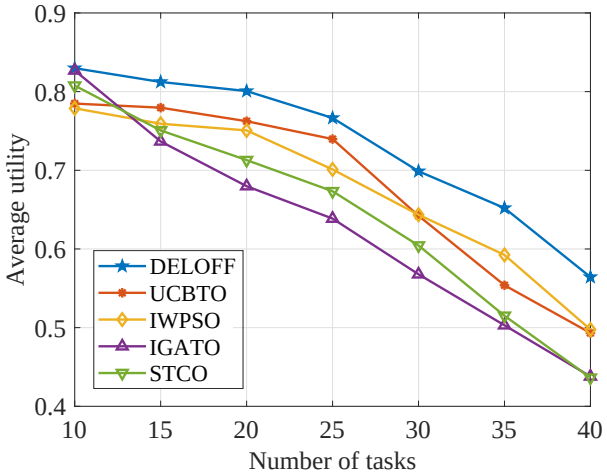
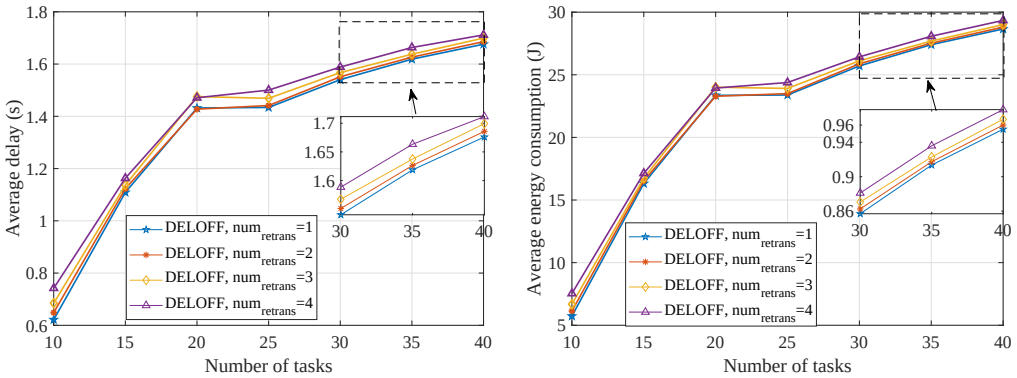


Figure 10. Average utility versus the number of tasks.



(**a**) Average delay. (**b**) Average energy consumption.

Figure 11. Average delay and energy consumption considering the maximum allowable number of retransmission times.

6.4. Performance Comparison with Varying Task Computation Demands

Figure 12 exhibits the performance comparison of the DELOFF and four other approaches with the increasing needs for computation resources, including the average delay, energy consumption, and utility, where the total number of UAVs is set to be 25. As depicted in Figure 12a,b, the average delay and energy consumption increase with the growth of the required computation resources. This is because high computation demands will consume more computing delay and energy, which certainly degrades the average utility shown in Figure 12c. The delay and energy consumption of our scheme raise much more slowly when the average CPU cycles required by tasks is less than 2.5, and the utility shows the opposite trend, correspondingly. It occurs because our scheme has the ability to choose the best offloading target for each UAV from the available action space through decentralized repetition learning. The UCBTO algorithm always presents a certain performance gap with the DELOFF. On the other side, the performance of IWPSO in terms of delay, energy consumption, and utility is very close to UCBTO, which is because UAVs

in IWPSO can converge to the relative optimal solution in some cases. Finally, based on the above observations, our proposed DELOFF outperforms four other mechanisms and reduces delay and energy consumption by up to 7.13% and 14.44%, respectively. At the same time, the average utility is improved by at least 11.97%.

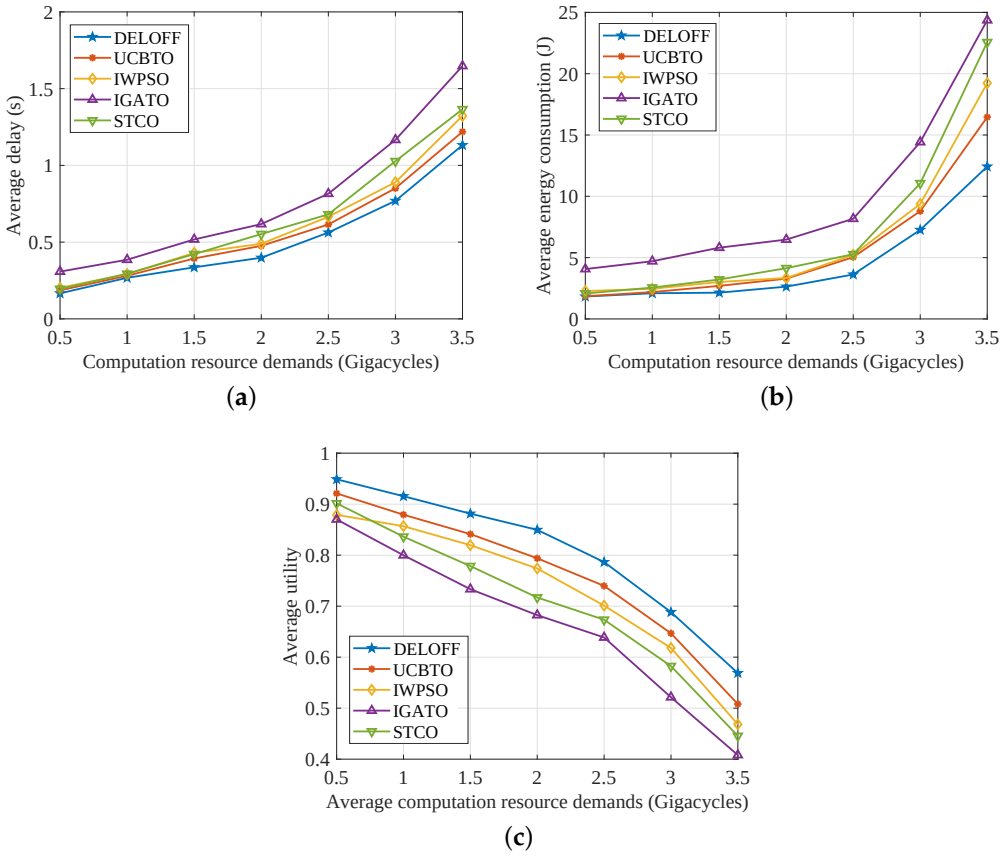


Figure 12. Impact of computation resource demands on offloading performance. (**a**) Average delay versus computation resource demands. (**b**) Average energy consumption versus computation resource demands. (**c**) Average utility versus computation resource demands.

6.5. Performance Comparison with Varying Task Data Size

The impact of the varying data size of tasks is demonstrated in Figure 13 in terms of average delay, average energy consumption, and average utility on a fixed number of UAVs (which is 25). In Figure 13a,b, the average delay and energy consumed by each UAV present a rising trend, because the larger the data size of tasks, the more communication delay and energy cost when tasks are offloaded. In this case, the average utility obtained by each UAV, depicted in Figure 13c, consequently decreases as the data size increases. Particularly, due to the efficient use of the offloading pre-screening mechanism and the exploitation of the best response of each UAV, our devised scheme provides the best performance results with the increment of the data size. Using the model of IGATO, optimal offloading strategies cannot be extensively derived. It is due to this that the IGATO may fail to deal with the offloading problem with constrained optimization. Additionally, in the scenario of increasing the data size of computation tasks, the performance discrepancy between UCBTO and IWPSO varies compared to that observed when the number of tasks changes. As depicted in Figure 13, as the task size increases, the IWPSO algorithm performs more poorly than the UCBTO algorithm in terms of delay, energy consumption, and utility. This can be attributed to the fact that, unlike scenarios where the number of tasks varies, maintaining a fixed number of tasks as the data size expands leads to a more complex solution space. As a result, the IWPSO scheme may face challenges in exploring the global

optimal solution and may become trapped near local optima. In general, as the data size increases, the average delay and energy consumption in our proposed DELOFF decreases by at least 6.58% and 26.18% by comparing with four other schemes, and the average utility can be optimized by up to 4.74%.

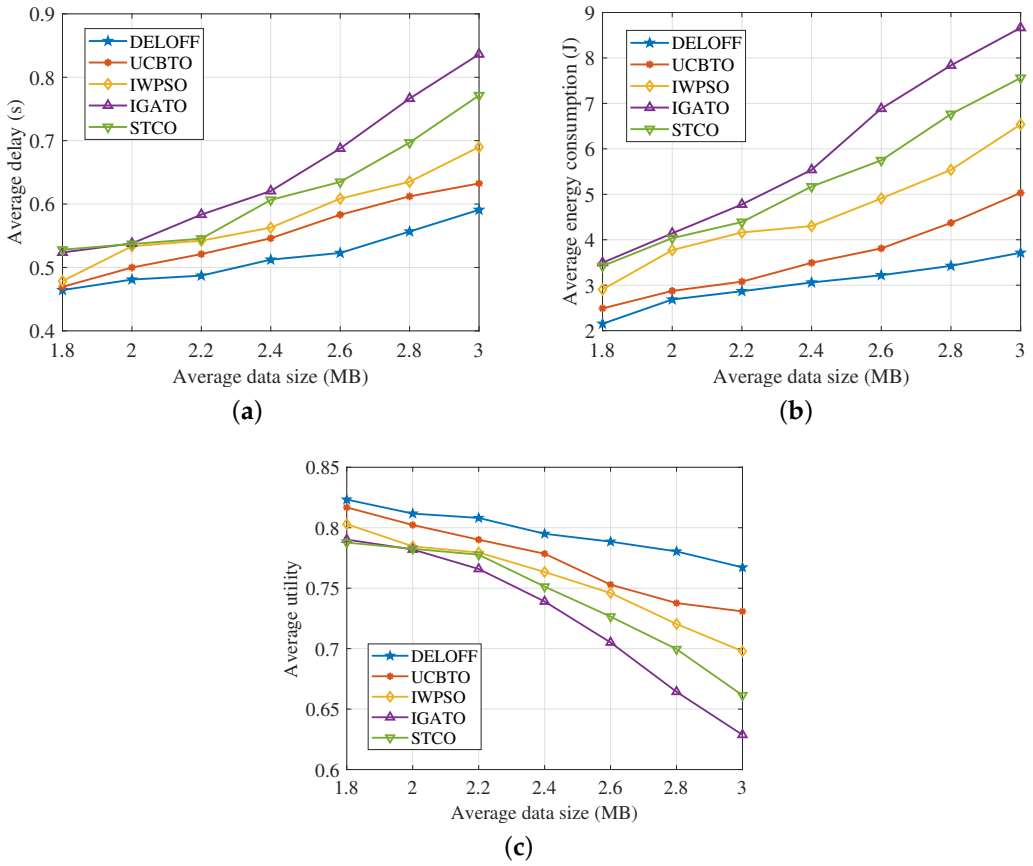
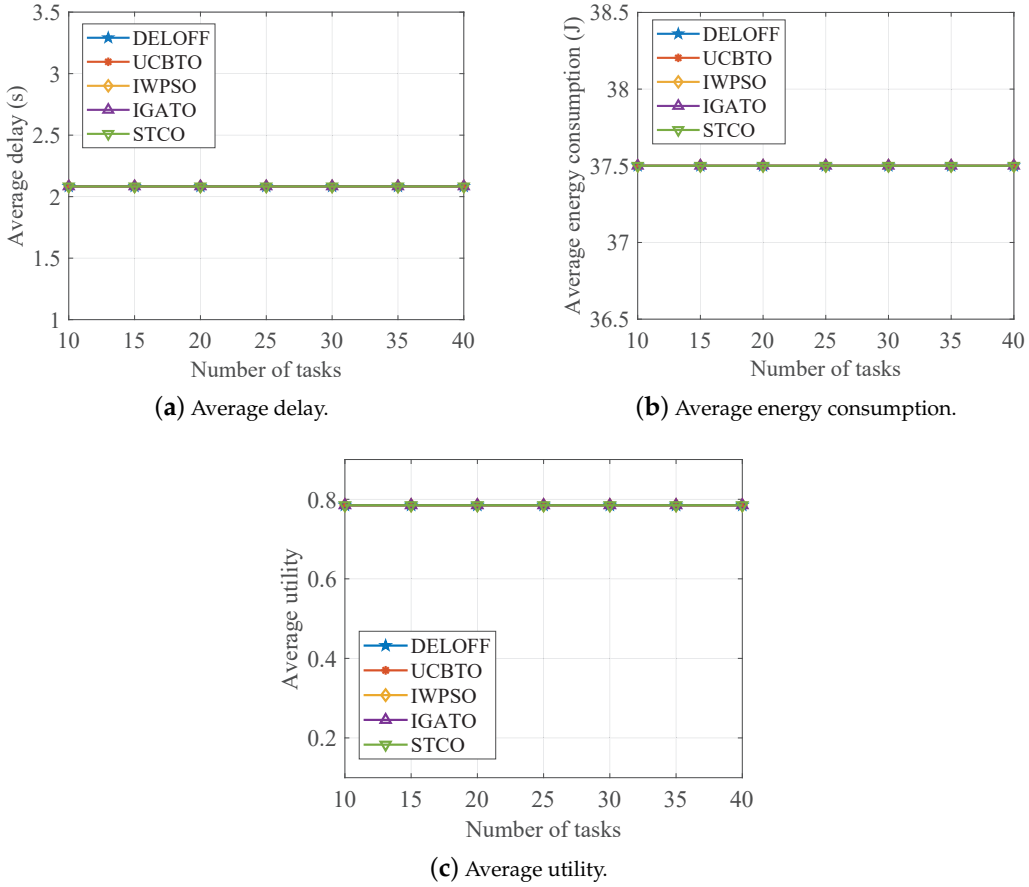


Figure 13. Impact of task data size on offloading performance. (**a**) Average delay versus data size. (**b**) Average energy consumption versus data size. (**c**) Average utility versus data size.

6.6. Discussion on UAV Hovering

We further investigate the scenario in which UAVs hover and wait for the completion of offloading, considering the energy consumption associated with hovering, as defined in Equation (14). Figure 14a,b demonstrate that the average delay and energy consumption exhibit no variation with the increasing number of tasks, resulting in a constant utility, depicted in Figure 14c. This observation indicates that all algorithms consistently opt for local computing to execute tasks. The rationale behind this is that the significant energy consumption required for the UAVs to hover and wait for offloading completion, which far exceeds the energy consumed during offloading transmission, so the UAVs choose to process tasks locally to avoid excessive energy consumption. Hence, it can be observed that it is unnecessary for UAVs to offload computation tasks while hovering, as this would result in excessive energy consumption for the UAVs. This would mean that, in most cases, the computation offloading of UAVs would be ineffective, as the UAVs, at this point, undergo offloading but cannot necessarily achieve the expected performance improvement. Instead, they would consume excessive energy by hovering. On the other hand, performing local computing on the UAVs would greatly deplete their limited onboard capabilities. Furthermore, this validates the effectiveness of the proposed DELOFF scheme as it takes into account the ability of UAVs to offload computation tasks in dynamic scenarios, allowing them to continue their scheduled operations while offloading. By offloading tasks in such a scenario, UAVs not only conserve battery life, but also process computation tasks in a

timely manner, significantly extending their onboard resources through the utilization of available network resources.



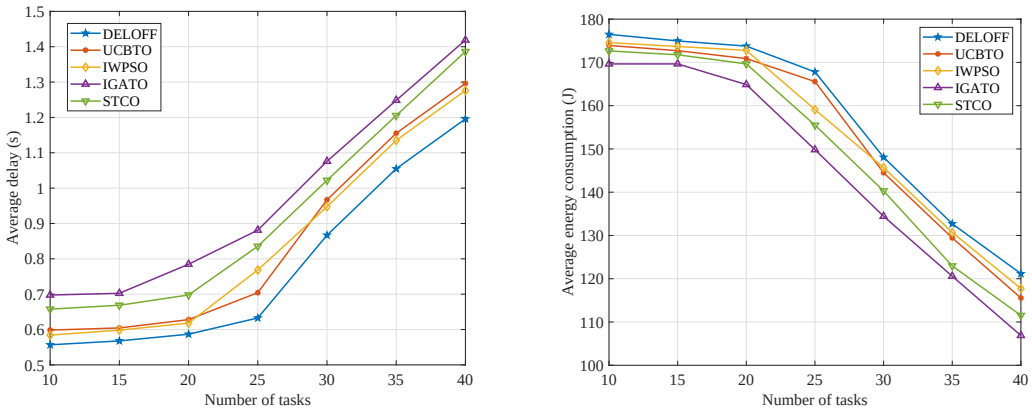
(**a**) Average delay.

(**b**) Average energy consumption.

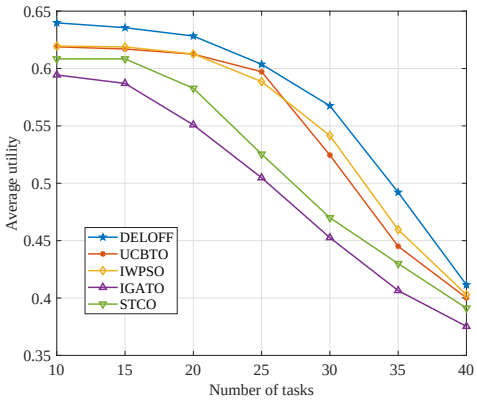
(**c**) Average utility.

Figure 14. Impact of UAV hovering on computation offloading performance.

In addition, we extend to investigate the impact of UAVs with time-sensitive tasks on the energy consumption for computation offloading in scenarios where the UAVs hover and wait for offloading to be completed. Specifically, the delay weighting factor $\delta_{\psi_h,d}$ is set to 0.9, indicating that the UAVs prioritize timely task processing and increase their sensitivity to the delay associated with task processing. As depicted in Figure 15, the generated average delay, energy consumption and, utility achieved by the UAVs concerning the number of tasks are recorded, respectively. From Figure 15a,c, it can be seen that our proposed DELOFF mechanism achieves the lowest delay and the highest utility. This demonstrates that DELOFF can adaptively trade off low task computation time for high energy costs, as shown in Figure 15b, in scenarios with high real-time task requirements. Conversely, it can be inferred that our algorithm proactively optimizes offloading to save energy under conditions of limited UAV energy and that the task does not have high time requirements. In this study, we conduct an investigation into the energy consumption associated with the hovering behavior of the UAV while waiting for offloading to be completed. We evaluate the influence of hovering energy through simulations and validate the effectiveness of the DELOFF proposed on this extended energy model. To further explore the applicability of the proposed offloading scheme, future research can incorporate more detailed energy modeling, as exemplified by the works referenced in [60,61].



(**a**) Average delay. (**b**) Average energy consumption.



(**c**) Average utility.

Figure 15. Impact of UAV hovering on time-sensitive task offloading.

7. Conclusions

In this paper, to provide massive access and differentiated service requests for UAVs with heterogeneous application tasks to execute, we study the computation offloading of multi-UAVs by extending the problem to complicated U2X-assisted heterogeneous networks, and propose a decentralized optimization framework to tackle the joint decision making associated with the selection of network access, computation mode, and offloading allocation for UAVs. Specifically, a fuzzy logic-based offloading pre-screening scheme is devised, which is managed on the UAV side for the purpose of adaptively identifying the available offloading targets that can achieve stable offloading performance during wireless connection. Further, by resorting to the multi-arm bandit learning framework, the dedicated exploration, assignment, and exploitation mechanisms are designed in the DELOFF proposed to search for the optimal solution for keeping utility maximization for UAVs via observing the bandit feedbacks locally without introducing extra information interaction. The comprehensive simulation results show that the proposed DELOFF demonstrates its effectiveness and superior performance over the mainstream schemes in various scenarios, and shows efficient convergence and adaptability. Consequently, the DELOFF can be applied broadly in the universal and scalable scenario for UAVs to perform compute-intensive applications and enhance operational efficiency. In future research, we will extend the investigation of the proposed DELOFF framework by deploying it in the FlockAI platform, which has proven effective in modeling and testing for UAV applications [62,63]. This deployment will enable us to explore the usage of resources, network overhead, and power consumption more accurately in different scenarios. By doing so, we aim to identify performance inefficiencies and potential trade-offs in UAV applications and task offloading. And, we will examine how different configurations and

computation offloading models impact the overall system. This approach will provide a comprehensive understanding of the framework's capabilities and its implications in practical settings. Furthermore, we will conduct real-world experiments and ensure that the experiments cover a wide range of characteristics to capture the diversity of scenarios and potential challenges. By varying parameters such as environmental conditions and input data characteristics, we aim to investigate the behavior of the methodology and its applicability to UAV offloading in different contexts.

Author Contributions: Conceptualization, A.Z. and H.L.; methodology and writing—original draft, A.Z. and M.M.; writing—review and editing, Z.Z. (Zongtan Zhou) and Z.Z. (Zhiwen Zeng). All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China under Grant U1913202.

Data Availability Statement: Data sharing not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Cui, Y.; Liu, F.; Jing, X.; Mu, J. Integrating sensing and communications for ubiquitous IoT: Applications, trends, and challenges. *IEEE Netw.* **2021**, *35*, 158–167.
2. Lucic, M.C.; Bouhamed, O.; Ghazzai, H.; Khanfor, A.; Massoud, Y. Leveraging UAVs to Enable Dynamic and Smart Aerial Infrastructure for ITS and Smart Cities: An Overview. *Drones* **2023**, *7*, 79.
3. Khan, M.A.; Ullah, I.; Alkhalifah, A.; Rehman, S.U.; Shah, J.A.; Uddin, M.I.; Alsharif, M.H.; Algarni, F. A Provable and Privacy-Preserving Authentication Scheme for UAV-Enabled Intelligent Transportation Systems. *IEEE Trans. Ind. Inform.* **2022**, *18*, 3416–3425. [CrossRef]
4. Zhu, A.; Zeng, Z.; Guo, S.; Lu, H.; Ma, M.; Zhou, Z. Game-theoretic robotic offloading via multi-agent learning for agricultural applications in heterogeneous networks. *Comput. Electron. Agric.* **2023**, *211*, 108017.
5. Zhang, Z.; Zhu, L. A Review on Unmanned Aerial Vehicle Remote Sensing: Platforms, Sensors, Data Processing Methods, and Applications. *Drones* **2023**, *7*, 398.
6. Zhu, A.; Guo, S.; Liu, B.; Ma, M.; Yao, J.; Su, X. Adaptive Multiservice Heterogeneous Network Selection Scheme in Mobile Edge Computing. *IEEE Internet Things J.* **2019**, *6*, 6862–6875. [CrossRef]
7. Ma, M.; Wang, Z. Distributed Offloading for Multi-UAV Swarms in MEC-Assisted 5G Heterogeneous Networks. *Drones* **2023**, *7*, 226.
8. Zhu, A.; Lu, H.; Guo, S.; Zeng, Z.; Zhou, Z. CollOR: Distributed collaborative offloading and routing for tasks with QoS demands in multi-robot system. *Ad Hoc Netw.* **2023**, *152*, 103311. [CrossRef]
9. McEnroe, P.; Wang, S.; Liyanage, M. A Survey on the Convergence of Edge Computing and AI for UAVs: Opportunities and Challenges. *IEEE Internet Things J.* **2022**, *9*, 15435–15459. [CrossRef]
10. Ma, M.; Zhu, A.; Guo, S.; Yang, Y. Intelligent network selection algorithm for multiservice users in 5G heterogeneous network system: Nash Q-learning method. *IEEE Internet Things J.* **2021**, *8*, 11877–11890.
11. Zhu, A.; Ma, M.; Guo, S.; Yu, S.; Yi, L. Adaptive multi-access algorithm for multi-service edge users in 5G ultra-dense heterogeneous networks. *IEEE Trans. Veh. Technol.* **2021**, *70*, 2807–2821.
12. Liu, H.; Niu, Z.; Du, J.; Lin, X. Genetic algorithm for delay efficient computation offloading in dispersed computing. *Ad Hoc Netw.* **2023**, *142*, 103109.
13. Bacanin, N.; Antonijevic, M.; Bezdan, T.; Zivkovic, M.; Venkatachalam, K.; Malebary, S. Energy efficient offloading mechanism using particle swarm optimization in 5G enabled edge nodes. *Clust. Comput.* **2023**, *26*, 587–598.
14. Ma, M.; Zhu, A.; Guo, S.; Wang, X.; Liu, B.; Su, X. Heterogeneous network selection algorithm for novel 5G services based on evolutionary game. *IET Commun.* **2020**, *14*, 320–330.
15. Zhu, A.; Ma, M.; Guo, S.; Yang, Y. Adaptive Access Selection Algorithm for Multi-Service in 5G Heterogeneous Internet of Things. *IEEE Trans. Netw. Sci. Eng.* **2022**, *9*, 1630–1644. [CrossRef]
16. Alqurashi, F.A.; Alsolami, F.; Abdel-Khalek, S.; Sayed Ali, E.; Saeed, R.A. Machine learning techniques in internet of UAVs for smart cities applications. *J. Intell. Fuzzy Syst.* **2022**, *42*, 3203–3226.
17. Zhu, A.; Lu, H.; Guo, S.; Zeng, Z.; Ma, M.; Zhou, Z. SyRoC: Symbiotic robotics for QoS-aware heterogeneous applications in IoT-edge-cloud computing paradigm. *Future Gener. Comput. Syst.* **2023**, *150*, 202–219. [CrossRef]
18. Li, H.; Wu, S.; Jiao, J.; Lin, X.H.; Zhang, N.; Zhang, Q. Energy-Efficient Task Offloading of Edge-Aided Maritime UAV Systems. *IEEE Trans. Veh. Technol.* **2023**, *72*, 1116–1126. [CrossRef]
19. Alfakih, T.; Hassan, M.M.; Gumaei, A.; Savaglio, C.; Fortino, G. Task Offloading and Resource Allocation for Mobile Edge Computing by Deep Reinforcement Learning Based on SARSA. *IEEE Access* **2020**, *8*, 54074–54084. [CrossRef]

20. Afrin, M.; Jin, J.; Rahman, A.; Gasparri, A.; Tian, Y.C.; Kulkarni, A. Robotic Edge Resource Allocation for Agricultural Cyber-Physical System. *IEEE Trans. Netw. Sci. Eng.* **2022**, *9*, 3979–3990. [CrossRef]
21. You, Q.; Tang, B. Efficient task offloading using particle swarm optimization algorithm in edge computing for industrial internet of things. *J. Cloud Comput.* **2021**, *10*, 1–11.
22. He, Y.; Xu, J.; Zheng, B.; Hu, J.; Xie, Y. Timing-Oriented Task Offloading Algorithms for Internet-of-Vehicles. *J. Circuits, Syst. Comput.* **2022**, *31*, 2250151. [CrossRef]
23. Zhang, D.; Li, X.; Zhang, J.; Zhang, T.; Gong, C. New Method of Task Offloading in Mobile Edge Computing for Vehicles Based on Simulated Annealing Mechanism. *J. Electron. Inf. Technol.* **2022**, *44*, 1–11.
24. Zhu, A.; Guo, S.; Ma, M.; Feng, H.; Liu, B.; Su, X.; Guo, M.; Jiang, Q. Computation Offloading for Workflow in Mobile Edge Computing Based on Deep Q-Learning. In Proceedings of the 2019 28th Wireless and Optical Communications Conference (WOCC), Beijing, China, 9–10 May 2019; pp. 1–5. [CrossRef]
25. Dai, M.; Su, Z.; Xu, Q.; Zhang, N. Vehicle assisted computing offloading for unmanned aerial vehicles in smart city. *IEEE Trans. Intell. Transp. Syst.* **2021**, *22*, 1932–1944. [CrossRef]
26. Xu, F.; Yang, W.; Li, H. Computation offloading algorithm for cloud robot based on improved game theory. *Comput. Electr. Eng.* **2020**, *87*, 106764. . [CrossRef]
27. Nguyen, A.C.; Pamuklu, T.; Syed, A.; Kennedy, W.S.; Erol-Kantarci, M. Reinforcement Learning-Based Deadline and Battery-Aware Offloading in Smart Farm IoT-UAV Networks. In Proceedings of the ICC 2022—IEEE International Conference on Communications, Seoul, Republic of Korea, 16–20 May 2022; pp. 189–194. [CrossRef]
28. Chen, X.; Zhang, J.; Lin, B.; Chen, Z.; Wolter, K.; Min, G. Energy-Efficient Offloading for DNN-Based Smart IoT Systems in Cloud-Edge Environments. *IEEE Trans. Parallel Distrib. Syst.* **2022**, *33*, 683–697. [CrossRef]
29. Chen, M.; Wang, T.; Zhang, S.; Liu, A. Deep reinforcement learning for computation offloading in mobile edge computing environment. *Comput. Commun.* **2021**, *175*, 1–12. [CrossRef]
30. Zhang, X.; Cao, Y. Mobile Data Offloading Efficiency: A Stochastic Analytical View. In Proceedings of the 2018 IEEE International Conference on Communications Workshops (ICC Workshops), Kansas City, MO, USA, 20–24 May 2018; pp. 1–6. [CrossRef]
31. Zhang, X.; Wang, J. Joint heterogeneous statistical-QoS/QoE provisionings for edge-computing based WiFi offloading over 5G mobile wireless networks. In Proceedings of the 2018 52nd Annual Conference on Information Sciences and Systems (CISS), Princeton, NJ, USA, 21–23 March 2018; pp. 1–6. [CrossRef]
32. Yang, G.; Hou, L.; He, X.; He, D.; Chan, S.; Guizani, M. Offloading Time Optimization via Markov Decision Process in Mobile-Edge Computing. *IEEE Internet Things J.* **2021**, *8*, 2483–2493. [CrossRef]
33. Wang, K.; Wang, X.; Liu, X. A high reliable computing offloading strategy using deep reinforcement learning for iovs in edge computing. *J. Grid Comput.* **2021**, *19*, 1–15.
34. Nguyen, T.; Katila, R.; Gia, T.N. An advanced internet-of-drones system with blockchain for improving quality of service of search and rescue: A feasibility study. *Future Gener. Comput. Syst.* **2023**, *140*, 36–52. [CrossRef]
35. Pliatsios, D.; Sarigiannidis, P.; Lagkas, T.D.; Argyriou, V.; Boulogeorgos, A.A.A.; Baziana, P. Joint Wireless Resource and Computation Offloading Optimization for Energy Efficient Internet of Vehicles. *IEEE Trans. Green Commun. Netw.* **2022**, *6*, 1468–1480. [CrossRef]
36. Sacco, A.; Esposito, F.; Marchetto, G.; Montuschi, P. Sustainable Task Offloading in UAV Networks via Multi-Agent Reinforcement Learning. *IEEE Trans. Veh. Technol.* **2021**, *70*, 5003–5015. [CrossRef]
37. Guo, S.; Xiao, B.; Yang, Y.; Yang, Y. Energy-efficient dynamic offloading and resource scheduling in mobile cloud computing. In Proceedings of the IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications, San Francisco, CA, USA, 10–14 April 2016; pp. 1–9.
38. Lu, W.; Zhang, X. Computation Offloading for Partitionable Applications in Dense Networks: An Evolutionary Game Approach. *IEEE Internet Things J.* **2022**, *9*, 20985–20996. [CrossRef]
39. Hu, M.; Xie, Z.; Wu, D.; Zhou, Y.; Chen, X.; Xiao, L. Heterogeneous edge offloading with incomplete information: A minority game approach. *IEEE Trans. Parallel Distrib. Syst.* **2020**, *31*, 2139–2154. [CrossRef]
40. Liao, H.; Mu, Y.; Zhou, Z.; Sun, M.; Wang, Z.; Pan, C. Blockchain and Learning-Based Secure and Intelligent Task Offloading for Vehicular Fog Computing. *IEEE Trans. Intell. Transp. Syst.* **2021**, *22*, 4051–4063. [CrossRef]
41. Hamdi, M.; Ben Hamed, A.; Yuan, D.; Zaied, M. Energy-Efficient Joint Task Assignment and Power Control in Energy-Harvesting D2D Offloading Communications. *IEEE Internet Things J.* **2022**, *9*, 6018–6031. [CrossRef]
42. Dai, X.; Xiao, Z.; Jiang, H.; Alazab, M.; Lui, J.C.S.; Dustdar, S.; Liu, J. Task Co-Offloading for D2D-Assisted Mobile Edge Computing in Industrial Internet of Things. *IEEE Trans. Ind. Inform.* **2023**, *19*, 480–490. [CrossRef]
43. Zeng, Y.; Xu, J.; Zhang, R. Energy Minimization for Wireless Communication With Rotary-Wing UAV. *IEEE Trans. Wirel. Commun.* **2019**, *18*, 2329–2345. [CrossRef]
44. Reddy, S.; Panwar, L.K.; Panigrahi, B.K.; Kumar, R. Computational intelligence for demand response exchange considering temporal characteristics of load profile via adaptive fuzzy inference system. *IEEE Trans. Emerg. Top. Comput. Intell.* **2017**, *2*, 235–245. [CrossRef]
45. Pekaslan, D.; Wagner, C.; Garibaldi, J.M. ADONiS-Adaptive Online Nonsingleton Fuzzy Logic Systems. *IEEE Trans. Fuzzy Syst.* **2020**, *28*, 2302–2312. [CrossRef]

46. Fu, S.; Zhang, Y.; Ceriotti, M.; Jiang, Y.; Packeiser. Modeling packet loss rate of IEEE 802.15.4 links in diverse environmental conditions. In Proceedings of the 2018 IEEE Wireless Communications and Networking Conference (WCNC), Barcelona, Spain, 15–18 April 2018; pp. 1–6. [CrossRef]
47. Sadjadi, E. On the Monotonicity of Smooth Fuzzy Systems. *IEEE Trans. Fuzzy Syst.* **2020**, *29*, 3947–3952. [CrossRef]
48. Athanasiadou, G.E.; Fytampanis, P.; Zarbouti, D.A.; Tsoulos, G.V.; Gkonis, P.K.; Kaklamani, D.I. Radio network planning towards 5G mmWave standalone small-cell architectures. *Electronics* **2020**, *9*, 339. [CrossRef]
49. Garroppo, R.G.; Volpi, M.; Nencioni, G.; Wadatkar, P.V. Experimental Evaluation of Handover Strategies in 5G-MEC Scenario by using AdvantEDGE. In Proceedings of the 2022 IEEE International Mediterranean Conference on Communications and Networking (MeditCom), Athens, Greece, 5–8 September 2022; pp. 286–291.
50. Pandey, K.; Arya, R. Lyapunov optimization machine learning resource allocation approach for uplink underlaid D2D communication in 5G networks. *IET Commun.* **2022**, *16*, 476–484. [CrossRef]
51. Liu, J.; Zhou, A.; Liu, C.; Zhang, T.; Qi, L.; Wang, S.; Buyya, R. Reliability-Enhanced Task Offloading in Mobile Edge Computing Environments. *IEEE Internet Things J.* **2022**, *9*, 10382–10396. [CrossRef]
52. Ali, M.A.; Zeng, Y.; Jamalipour, A. Software-defined coexisting UAV and WiFi: Delay-oriented traffic offloading and UAV placement. *IEEE J. Sel. Areas Commun.* **2020**, *38*, 988–998. [CrossRef]
53. Zhao, S.; Feng, Y.; Yu, G. D2D communication channel allocation and resource optimization in 5G network based on game theory. *Comput. Commun.* **2021**, *169*, 26–32. [CrossRef]
54. Ghaseminajm, F.; Alsmadi, M.; Tubail, D.; Ikki, S.S. RIS-Aided Mobile Localization Error Bounds Under Hardware Impairments. *IEEE Trans. Commun.* **2022**, *70*, 8331–8341. [CrossRef]
55. Wu, B.; Chen, T.; Yang, K.; Wang, X. Edge-Centric Bandit Learning for Task-Offloading Allocations in Multi-RAT Heterogeneous Networks. *IEEE Trans. Veh. Technol.* **2021**, *70*, 3702–3714. [CrossRef]
56. Deng, X.; Sun, Z.; Li, D.; Luo, J.; Wan, S. User-centric computation offloading for edge computing. *IEEE Internet Things J.* **2021**, *8*, 12559–12568. [CrossRef]
57. Zhu, A.; Wen, Y. Computing Offloading Strategy Using Improved Genetic Algorithm in Mobile Edge Computing System. *J. Grid Comput.* **2021**, *19*, 1–12. [CrossRef]
58. Zhang, K.; Gui, X.; Ren, D.; Li, D. Energy-Latency Tradeoff for Computation Offloading in UAV-Assisted Multiaccess Edge Computing System. *IEEE Internet Things J.* **2021**, *8*, 6709–6719. [CrossRef]
59. Malik, A.W.; Rahman, A.U.; Ali, M.; Santos, M.M. Symbiotic robotics network for efficient task offloading in smart industry. *IEEE Trans. Ind. Inform.* **2020**, *17*, 4594–4601. [CrossRef]
60. Trihinas, D.; Agathocleous, M.; Avogian, K. Composable energy modeling for ml-driven drone applications. In Proceedings of the 2021 IEEE International Conference on Cloud Engineering (IC2E), San Francisco, CA, USA, 4–8 October 2021; pp. 231–237.
61. Marins, J.L.; Cabreira, T.M.; Kappel, K.S.; Ferreira, P.R. A closed-form energy model for multi-rotors based on the dynamic of the movement. In Proceedings of the 2018 VIII Brazilian Symposium on Computing Systems Engineering (SBESC), Salvador, Brazil, 5–8 November 2018; pp. 256–261.
62. Trihinas, D.; Agathocleous, M.; Avogian, K.; Katakis, I. FlockAI: A Testing Suite for ML-Driven Drone Applications. *Future Internet* **2021**, *13*, 317. [CrossRef]
63. Trihinas, D.; Agathocleous, M.; Avogian, K. Demo: FlockAI—A Framework for Rapidly Testing ML-Driven Drone Applications. In Proceedings of the 2022 IEEE 42nd International Conference on Distributed Computing Systems (ICDCS), Bologna, Italy, 10–13 July 2022; pp. 1318–1321. [CrossRef]