IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 21, 2019, accepted March 5, 2019, date of publication March 25, 2019, date of current version April 5, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2905249

Energy- and Latency-Aware Hybrid Offloading Algorithm for UAVs

ABDELHAMIED A. ATEYA[1,2], (Member, IEEE), AMMAR MUTHANNA[2,3], RUSLAN KIRICHEK[2], MOHAMMAD HAMMOUDEH[4], (Senior Member, IEEE), AND ANDREY KOUCHERYAVY[2]

[1]Electronics and Communications Engineering, Zagazig University, Zagazig 44519, Egypt
[2]St. Petersburg State University of Telecommunication, 193232 St. Petersburg, Russia
[3]Peoples' Friendship University of Russia (RUDN University), 117198 Moscow, Russia
[4]Faculty of Science and Engineering, Manchester Metropolitan University, Manchester M15 6BH, U.K.

Corresponding author: Abdelhamied A. Ateya (a_ashraf@zu.edu.eg)

The publication has been prepared with the support of the "RUDN University Program 5-100."

ABSTRACT One of the most promising use cases of 5G/IMT2020 is the unmanned aerial vehicle (UAV). Due to their small size, the UAVs are resource constraint devices. To this end, this paper proposes an offloading algorithm for UAVs to assist in the execution of computationally intensive tasks. The proposed algorithm provides two UAV offloading methods. The first offloading method is the air-offloading, where a UAV can offload its computing tasks to nearby UAVs that have available computing and energy resources. The second offloading method is the ground-offloading, which enables the offloading of tasks to an edge cloud server from the multi-level edge cloud units connected to ground stations. The proposed algorithm is energy- and latency-aware, i.e., it selects the execution device and the offloading method based on the latency and energy constraints. The intensive algorithm simulation over reliable conditions for various scenarios with different cases for each scenario is conducted and results are presented.

INDEX TERMS UAV, offloading, latency, energy, 5G, MEC.

I. INTRODUCTION

Unmanned Aerial Vehicles (UAVs), e.g., drones, have gained increasing interest in recent years [1], [2]. With the near release of fifth generation cellular system (5G), UAVs are expected to have many applications. These applications vary from simple environmental monitoring to the complex high security military applications [3]–[5]. There are many challenges associated with the development of UAV networks and applications. These challenges include [6]–[8]: Trajectory or path planning, collision avoidance, mobility control, cost, security, data offloading, energy consumption, latency and compatibility with existing systems and cellular networks.

Part of these challenges is associated with the limited capabilities of UAVs, due to their small size, e.g., micro-drones, required for many applications [3]. Applications with computationally intensive tasks and applications, e.g., image- or video-based, require high processing and energy resources, which affect real-time operation and life-time of an UAV system or even cause task blocking. In order to prolong the life time of the UAV and overcome battery constraints, energy resources should be used carefully. One way to improve applications timeliness and reduce energy consumption is offloading computation to other devices in the UAV network with available resources [9], [10]. These devices may be in the air tire or on the ground [11]. For UAVs, there are two possible methods for data offloading, air-offloading and ground-offloading. An UAV can offload its computing tasks to nearby UAVs with available computing and energy resources [12] or offload the computing tasks to ground stations connected to cloud servers [13].

Recent technologies that will be deployed for 5G can assist in the deployment of UAVs [14]. These technologies include Mobile Edge Computing (MEC), Software Defined Networking (SDN) and Network Function Virtualization (NFV). MEC enables the cloud computing capabilities at the edge of the Radio Access Network (RAN), one communication hop away from the end user; and thus, reduces the computation latency [15], [16]. With the deployment of MEC, UAVs can offload their computing tasks efficiently to edge cloud servers. MEC servers handle the offloaded computing tasks and transfer results to the corresponding UAV via

The associate editor coordinating the review of this manuscript and approving it for publication was Muhammad Imran.

the appropriate link [17], [18]. Furthermore, MEC servers achieve higher efficiency in terms of latency for the offloaded tasks, as the servers are located close to the UAVs; approximately one communication hop away [19], [20]. Whereas SDN can be deployed to manage and control resources allocation for UAVs. Additionally, SDN controller can be used to optimize the utilization of MEC servers for handling UAV applications [21].

In this work, an energy- and latency-aware offloading algorithm is developed for UAVs to enable offloading of computing tasks, either through an air-offloading or a ground-offloading. UAVs employ a decision engine that decides whether to handle the task locally at the device or offload it to either air or ground devices. The ground offloading process is based on our algorithm introduced in [22]. This offloading algorithm has been developed for our multilevel MEC system presented in [23]. The proposed algorithm comprises three main sub routines to prolong the lifetime of UAV networks by saving resources through task offloading. Furthermore, the proposed algorithm reduces the probability of task blocking due to lack of resources and also reduces the end-to-end latency of handling a computing task. This algorithm is developed mainly for UAVs with image and video based applications, especially, military applications that requires UAVs with small sizes.

Section 2 introduces the related works to our proposed offloading algorithm. In Section 3, the offloading algorithm details are presented. Section 4 presents the simulation setup and results analysis. Section 5 concludes the paper.

II. RELATEDWORKS

In this section, recent related studies to the proposed offloading algorithms for UAV networks are reviewed. The advantages and limitations of each work are described. Furthermore, the novelty of our proposed algorithm is compared to these studies..

Luo *et al.* [24] proposed a framework for UAVs deployed for disaster related applications. The framework offloads the sensed data, e.g., video data, to a remote cloud unit. The system employs a client-server mechanism, where the client is hosted on the UAV and the server is hosted in the remote cloud unit. The client is responsible for video acquisition, data scheduling through a context-aware video scheduler and data offloading. The server receives the offloaded data and provides the computing resources required for handling the received data. The main limitation with this framework is latency as the data is offloaded to remote cloud rather than an edge cloud located closer to the UAV network.

Lynskey [25] developed a MEC based algorithm that maximizes the probability of offloading by optimizing the computing resources used to handle a certain offloaded task. The work has been developed mainly to process the offloaded images from UAVs. The introduced algorithm has been implemented at MEC servers to find the optimal combination of offloaded tasks that can be processed based on the MEC server capacity. This work does not consider the offloading process; it considers the execution process and only ground-offloading.

Zhou *et al.* [26] introduced four use cases of UAVs, which are supported by cloud and edge computing. The work mainly defines the methods of interaction between UAVs and ground devices, including heterogeneous computing devices. The authors considered a case study for UAVs with the MEC structure, this scenario is the use of UAVs to assist traffic and connectivity for road topologies. The work mainly considers using UAVs to assist massive sensor networks and dense IoT networks. It shares the similarity of deploying MEC technology with our proposed work; however, the MEC system deployed for our proposed system is different in the sense that it is deployed in multilevel structure. Furthermore, this work considers a structure for a certain application only.

Valentino *et al.* [12] have developed an offloading algorithm that offloads computation tasks to nearby UAVs. The proposed system assumes that UAVs are deployed in clusters; each cluster consists of a group of nearby UAVs that mostly achieve common tasks. Each UAV cluster has a cluster head that manages and controls other cluster UAVs members. The proposed algorithm enables UAV to offload their tasks to other clusters with available resources, if the current cluster has no available computing resources. The cluster head only decides whether to execute the task locally (i.e., in the current cluster) or offload it to a nearby cluster with available resources. Furthermore, the offloading of computing tasks between clusters distributes the computing tasks among UAVs, which achieves higher energy efficiency and prolongs the UAV's lifetime. Each cluster head has the responsibility to search nearby clusters for available computing and energy resources. This algorithm can only be deployed for dense UAV networks, with the consideration of clustering. This work did not consider the ground offloading; it only considered the air-offloading and only for clusters of UAVs.

Jung *et al.* [27] developed an adaptive offloading algorithm for UAVs that uses multipath TCP for the offloading process. This work considers only the ground offloading, where the adaptive offloading algorithm selects the best ground edge server that can host the computing tasks. The system is developed mainly for mission critical applications and to prolong the UAV lifetime. The data is offloaded from the UAV to the ground server via multiple TCP connections. The main advantage with this work is the consideration of UAV's mobility while task offloading and handling. Moreover, the handoff between base stations while handling a task, if the UAV moves between two coverage regions, has been considered.

An SDN supported algorithm was proposed in [28] to allocate computing and processing resources to UAVs hovering over a certain geographical area and are connected to one base station. The SDN controller deploys a greedy algorithm that accepts the offloaded applications and selects the optimal cloud server with available resources that achieves the required QoS latency of the offloaded task. The cloud servers deployed in the system are heterogeneous;

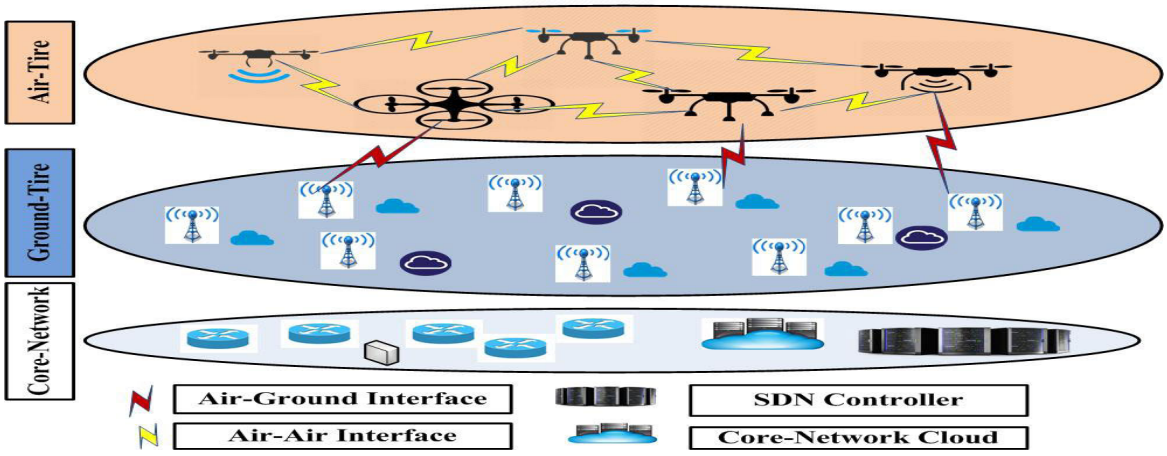

FIGURE 1. The end-to-end system structure of multilevel MEC based UAVs system.

edge cloud servers, distributed cloud servers connected to some buildings and remote cloud servers in the Internet. Each base station hosts an SDN based application that employs the greedy based algorithm to select the best cloud server for the current UAV's task. This work has been developed for heterogeneous UAVs with different computing and energy capabilities. One main disadvantage of this algorithm is the neglect of UAV's displacement, while handling a task. The base station assumes that the UAV's are stationary while receiving and processing a computing task. The main advantage with this algorithm is that it is implemented by the SDN controller, which makes use of the great benefits of the SDN based networks. Our proposed work differs from this work, as our ground system employs a multilevel based edge computing system.

The novelty of our proposed algorithm, compared to the above described works comes from the deployment of hybrid offloading. The UAV can toggle between two local execution and two offloading methods. It deploys a decision engine to select the offloading method that best achieves energy and latency efficiency. Moreover, both air- and ground-offloading processes differ from the previously introduced methods in the previous reviewed works. The ground-offloading is carried out over a multilevel MEC system structure connected to the ground stations. This multilevel structure has been developed in [23] and [29] and the offloading algorithm for such structure has been developed in [22].

III. HYBRID OFFLOADING ALGORITHM FOR UAVS

Recent applications and use cases of UAVs require the deployment of lightweight and small size UAVs, especially, for video-based surveillance applications [30]. Most of UAV applications required computing and power capabilities to collect and analyze application data. For video-based applications, the UAV has to process high resolution images, which requires high computing and energy capabilities than that available in commercial UAVs [31]. To overcome the computing and power limitations of UAVs, computationally expensive tasks should be offloaded to nearby units with available computing resources. In this section, we present a latency- and energy-aware MEC based offloading algorithm for UAVs to prolong the operation lifetime of UAVs and achieve shorter latency for heterogeneous UAV applications.

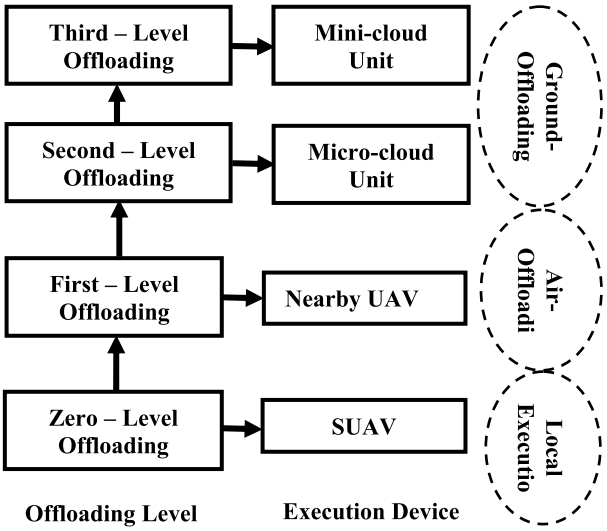

FIGURE 2. Main offloading levels.

A. SYSTEM STRUCTURE

The proposed system comprises the air tire and the ground tire, as illustrated in Figure 1. The air tire consists of the UAVs that work together or individually to provide certain services. The ground tire contains the ground base stations and the connected MEC servers to provide the computing capabilities at the edge of the RAN near to the end user.

The proposed offloading algorithm consists of several offloading levels which are illustrated in Figure 2. Each level represents an execution device with computing and energy resources. If the available resources of an UAV can handle a computing task within the required QoS latency, the task is processed locally at the UAV with no offloading. This represents the zero offloading level, which is suitable for simple tasks that require low computing and energy resources.

The first offloading level is where there offloading is directed to available nearby UAVs – referred to as air-offloading. When the onboard computing resources and energy of the UAV are not enough to handle a certain task, the UAV can offload the workload to one or more of the nearby UAVs that has available computing and energy resources. UAV can discover the nearby UAVs and ask for available resources for a certain task for an air-offloading process.

The higher offloading level includes the heterogeneous edge cloud units connected to ground base stations. This offloading level is referred to as the ground-offloading. In [22], we have introduced a multilevel MEC system to assist latency sensitive applications, e.g., AR/VR and Tactile Internet. We use this structure as the ground MEC system to assist offloading. The third offloading level is represented by the micro-cloud edge units, which provide computing resources at the edge of the RAN; one communication hop away from the target UAV. The fourth offloading level includes the mini-cloud edge units, which control and manage micro-cloud units. Mini-cloud units have higher computing and energy resources.

Each UAV deploys a decision engine, where our proposed offloading selecting algorithm is implemented. The decision engine decides, whether the current task can be executed locally or it need to be offloaded to an appropriate level of offloading. Moreover, the edge computing units, i.e., micro- and mini-clouds, employ local decision engine to decide the execution location of a certain task, based on the available computing and energy resources. Figure 3 illustrates the structure of the UAVs and the edge computing units.

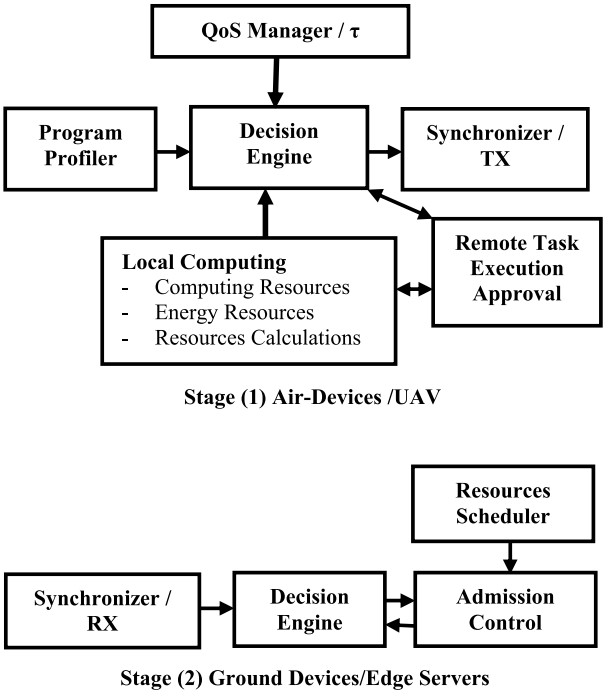

FIGURE 3. Main structure of the system stages.

The system is assumed to deploy heterogeneous types of UAVs with heterogeneous capabilities, in terms of computing capabilities and energy capabilities. These UAVs are deployed over a certain geographical regions and controlled through the nearest cellular base stations. UAVs are deployed for heterogeneous applications and tasks (e.g. surveillance, military application and cell coverage provide (drone- base station). In order to distinguish the UAV with the task required to be executed, it is referred to as the source UAV (SUAV).

B. LATENCY-AWARE AND ENERGY-AWARE OFFLOADING ALGORITHM

1) ASSUMPTIONS

In this paper, we make the following assumptions:

1. UAVs have heterogeneous power capabilities
2. UAVs have heterogeneous computing capabilities
3. UAVs run different applications with various computational and QoS needs
4. Full offloading is considered only, no partial offloading is considered; and
5. Queuing delay is neglected.

2) ANNOTATION

In this part, we define all parameters that will be used in the following sections, while introducing the offloading algorithm and corresponding mathematical equations. Table 1 introduces the notation of variables and parameters. Various steps of the proposed algorithms are indicated in algorithm1, 2 and 3. Algorithm 1 represents the steps for local processing and offloading decisions. Algorithm 2 represents the main steps for air-offloading process and algorithm 3 indicates the steps for ground-offloading process.

C. OFFLOADING MODEL

The offloading process is performed by the SUAV that decides whether to process the task locally or offload it. The SUAV first calculates the total size of task data, Z, and the total number of CPU cycles, N_{CYC}, required to process the current task. SUAV extracts this data through the program profiler and then forwards this data to the decision engine of SUAV. The decision engine also receives information about the maximum allowable latency $\bar{C}$ for handling the current task that supports the QoS, from the QoS manager.

The SUAV's decision engine calculates the total time required to execute the current task locally at the SUAV (T_{SUAV}), see Eq. (1), based on the current available resources. The decision engine checks the binary time decision variable of offloading $D_{T-\mathrm{off}}$, see Eq. (3), by comparing T_{SUAV} with the latency QoS time $\bar{C}$.

$$T_{SUAV} = \frac{N_{CYC}}{R_{SUAV}}, \quad R_{SUAV} \in f_{SUAV} \tag{1}$$

$$N_{CYC} = Z.K \tag{2}$$

$$D_{T-off} = I\left(T_{SUAV}, \tau\right) = \begin{cases} 0 & IF\ (T_{SUAV} \leq \tau) \\ 1 & IF\ (T_{SUAV} > \tau) \end{cases} \tag{3}$$

TABLE 1. Key annotation.

Notation	Description
Z	Total size of input data of a task in bits, which includes the program codes and other corresponding data
K	The number of CPU cycles required to process one bit of task input data
$\bar{C}$	Maximum allowable latency of a task *i*, which set in a way that supports QoS
N_{CYC}	The total number of required CPU cycles to process a task of length Z (bits)
R_{SUAV}	The processing resources of SUAV allocated for a task *i*
R_{UAV}	The processing resources of UAV allocated for executing a task *i*
R_{Micro}	The processing resources of micro-cloud edge server allocated for an offloaded task
R_{Mini}	The processing resources of mini-cloud edge server allocated for an offloaded task
δ_{SUAV}	Energy consumption per CPU cycle for workload execution at SUAV
δ_{UAV}	Energy consumption per CPU cycle for workload execution at UAV
δ_{Micro}	Energy consumption for one CPU cycle of micro-cloud edge server
δ_{Mini}	Energy consumption for one CPU cycle of mini-cloud edge server
T_{SUAV}	Total execution time of a task *i*, when the task is executed locally at the SUAV
$T_{ex\text{-}UAV}$	Total execution time of a certain task, executed at the UAV
T_{UAV}	Total time delay required for handling a task at the UAV
$T_{ex\text{-}Micro}$	Total execution time of a task, when the task is executed at the micro-cloud unit
T_{Micro}	Total time delay required to handle a task at micro-cloud unit
$T_{ex\text{-}Mini}$	Total execution time of a task, when the task is executed at the mini-cloud unit
T_{Mini}	Total time delay required to handle a task at Mini-cloud unit
E_{SUAV}	Energy consumption for executing a task *i* at the SUAV
E_{UAV}	Energy consumption for executing an offloaded task
E_{Micro}	Energy consumption for handling an offloaded task at micro-cloud edge server
E_{Mini}	Energy consumption for handling an offloaded task at mini-cloud edge server
E_{th}	Threshold level of energy of UAV
$E_{th\text{-}Micro}$	Threshold level of energy of micro-cloud edge server
$E_{th\text{-}Mini}$	Threshold level of energy of mini-cloud edge server
$E_{R\text{-}SUAV}$	The remaining energy of SUAV after task execution
$E_{R\text{-}UAV}$	The remaining energy of UAV after executing the offloaded task
$E_{R\text{-}Micro}$	The remaining energy of micro-cloud edge server after executing the offloaded task
$E_{R\text{-}Mini}$	The remaining energy of mini-cloud edge server after executing the offloaded task
$E_{C\text{-}SUAV}$	The current energy level of SUAV
$E_{C\text{-}UAV}$	The current energy level of UAV
$E_{C\text{-}Micro}$	The current energy level of micro-cloud edge server
$E_{C\text{-}Mini}$	The current energy level of mini-cloud edge server
$D_{air\text{-}off}$	Binary decision variable of air-offloading process decided by SUAV
$D_{ground\text{-}off}$	Binary decision variable of ground-offloading process decided by SUAV
$D_{E\text{-}off}$	Binary energy decision variable of offloading process decided by SUAV
$D_{T\text{-}off}$	Binary time decision variable of offloading process decided by SUAV
$D_{T\text{-}UAV\text{-}acc\text{-}off}$	Binary time decision variable of accepting task offloading decided by UAV
$D_{E\text{-}UAV\text{-}acc\text{-}off}$	Binary energy decision variable of accepting task offloading decided by UAV
$D_{T\text{-}Micro\text{-}acc\text{-}off}$	Binary time decision variable of accepting task offloading decided by micro-cloud edge server
$D_{E\text{-}Micro\text{-}acc\text{-}off}$	Binary energy decision variable of accepting task offloading decided by micro-cloud edge server
$D_{T\text{-}Mini\text{-}acc\text{-}off}$	Binary time decision variable of accepting task offloading decided by mini-cloud edge server
$D_{E\text{-}Mini\text{-}acc\text{-}off}$	Binary energy decision variable of accepting task offloading decided by mini-cloud edge server
D_{off}	Binary decision variable of offloading process decided by SUAV
I	Mode Indicator Variable
T_{comm}	Total communication latency between SUAV and nearby UAV
t_{pro}	Propagation latency
t_{trans}	Transmission latency
T_{tx}	Latency for transmitting task input data from the SUAV to the targeted nearby UAV
T_{rx}	Feedback time delay of computation results
C	Light speed ($3*10^8$ m/s)
$d_{x,y}$	Distance between source *x* and target *y*
R_b	Uplink achievable data transmission rate
ω	The system bandwidth
σ	Noise power at the receiver
h	Channel gain
P	Transmitting power of SUAV
P_S	Transmitting power of base station connected to Micro-cloud edge server
N	Total number of received respond messages / Total number of nearby UAVs
η_c	Channel efficiency
f_{SUAV}	Total processing resources of SUAV / CPU cycle frequency
f_{UAV}	Total processing resources of UAV / CPU cycle frequency
f_{Micro}	Total Micro-cloud edge server processing resources / CPU cycle frequency
f_{Mini}	Total Mini-cloud edge server processing resources / CPU cycle frequency

If the time decision is negative, the SUAV then checks the energy constraints by calculating the binary energy decision variable of offloading D_{E-off}. The binary energy decision

variable is calculated by comparing the remaining energy of the energy source on the UAV, E_{R-SUAV}, after consuming energy for handling the current task, with the threshold energy level of SUAV E_{th}. The threshold energy level represents the minimum level of energy of SUAV, after which the energy level of UAV is mentioned to be critical and cannot support handling computing tasks.

$$E_{SUAV} = N_{CYC}\delta_{SUAV} \tag{4}$$

$$E_{R-SUAV} = E_{C-SAUV} - E_{SUAV} \tag{5}$$

$$D_{E-off} = I\left(E_{R-SUAV}, E_{th}\right) = \begin{cases} 1 & IF(E_{R-SUAV} \leq E_{th}) \\ 0 & IF(E_{R-SUAV} > E_{th}) \end{cases} \tag{6}$$

Algorithm 1 Latency- and Energy-Aware Offloading Algorithm for SUAV

Input: $\mathbb{C}$, N_{CYC} and E_{th}
Output: D_{T-off} and D_{E-off}
Initialize $\mathbb{C}$, E_{th}
Calculate Z, K
Calculate T_{SUAV}
If ($T_{SUAV} \leq \mathbb{C}$)
  $D_{T-off} = 0$
  Calculate E_{SUAV}, E_{R-SUAV}
  If ($E_{R-SUAV} > E_{th}$)
    $D_{E-off} = 0$
    Handle task locally
  else
    $D_{E-off} = 1$
    Check air-offloading [Call algorithm 2]
  end if
else
  $D_{T-off} = 1$
  Check air-offloading [Call algorithm 2]
end if

If the remaining energy level of the SUAV, after consuming energy for the task execution, is larger than the threshold level of energy of the SUAV, the task is executed locally and there is no need to offload the task to any other devices. For this case, the energy decision binary variable is set to zero to indicate that there is no need for offloading. While, the energy decision variable is set to one if the remaining energy after executing current task will be less than the threshold energy level of SUAV. In this case, the offloading decision should be considered, either by offload to an air-device or to ground-device.

For a positive offloading decision, either time decision or energy decision, the SUAV should offload the computing task. The SUAV first, checks the possibility of air-offloading. This can be done by discovering the availability of computing resources of the nearby UAVs. If the conditions for the air-offloading are met, the SUAV takes the decision of air-offloading to the corresponding nearby UAV. This should be

Algorithm 2 Air-Offloading Algorithm

Input: N_{CYC}, R_{UAV} and E_{C-UAV}
Output: $D_{air-off}$
SUAV broadcasts request message
Nearby UAVs receives the request message
Nearby UAVs process the request:
Calculate T_{ex-UAV}, E_{UAV}, E_{R-UAV}
If ($E_{R-UAV} > E_{th}$)
  $D_{E-acc-off} = 1$
else
  $D_{E-acc-off} = 0$
end if
UAV transmits respond message
SUAV receives respond messages (N)
Count = 0
For (i = 1 : i ≤ N) do
  If ($D_{E\text{-}UAV}$-acc-off == 1)
    Calculate T_{UAV}(i)
    If (T_{UAV}(i) ≤ $\mathbb{C}$)
      $D_{air-off} = 1$
      Count ++
    end if
  else
    $D_{air-off} = 0$
  end if
End for
If ($D_{air-off}$ == 1)
  If (Count > 1)
    Select min(T_{UAV}(i))
  end if
  Offload the task to the appropriate nearby UAV
else
  Check ground-offloading [Call algorithm 3]
end if

happened, if there is a nearby UAV with sufficient energy and computing capabilities to handle the current task within the QoS latency time. However, if the air-offloading fails, due to the nonexistence of sufficient energy and computing resources among surrounding UAVs, the SUAV checks the ground-offloading.

1) AIR-OFFLOADING PROCESS

The air offloading process consists of two main sub-processes; the discovery process and the offloading process. Once the SUAV takes the decision of offloading, it first checks the availability of first level offloading. The SUAV looks for nearby UAVs to check if one of them has available resources to handle the task. For this purpose, the SUAV starts a discovery process, with the objective of discovering the surrounding field, whether it contains an UAV with available resources for task handling. Figure 4 illustrates the discovery process for the air-offloading process.

For the discovery process, the SUAV broadcasts a discovery message to all surrounding UAVs via an appropriate

Algorithm 3 Ground-Offloading Algorithm

Input: $\bar{C}$, N_{CYC}, R_{Micro} and R_{Mini}
Output: $D_{ground-off}$

```
Send request message of type I
Receive request message of type I
Calculate T_Micro, E_Micro, E_R−Micro
If (T_Micro ≤ C̄)
    D_T−acc−g−off = 1
    If (E_R−Micro > E_th−Micro)
        D_E−Micro−acc−g−off = 1
    else
        D_E−Micro−acc−g−off = 0
    end if
else
    D_T−Micro−acc−g−off = 0
end if
If (D_T−Micro−acc−g−off & D_E−Micro−acc−g−off == 1)
    D_ground−off = 1
    Offload the task to Micro-cloud
else
    Send request message of type II
    Receive request message of type II
    Calculate T_Mini, E_Mini, E_R−Mini
    if ( T_Mini ≤ C̄)
        D_T−Mini−acc−g−off = 1
        If (E_R−Mini > E_th−Mini)
            D_E−Mini−acc−g−off = 1
        else
            D_E−Mini−acc−g−off = 0
    end if
else
    D_T−Mini−acc−g−off = 0
end if
If (D_T−Mini−acc−g−off & D_E−Mini−acc−g−off == 1)
        D_ground−off = 1
        Offload the task to Mini-cloud
    else
        D_ground−off = 0
        Block the task
    end if
```

communication interface, e.g., WiFi [32]. The broadcasted message contains two main fields; the SUAV identification field and the task information field. Figure 5 illustrates the main fields and sub fields of the discovery message. The identification field contains the identification number of the SUAV (ID) and the longitude, latitude and the height of the current location of the SUAV. The task information field contains the information about the workload that required to be offloaded. This information includes; the total size of the computing data of the task to be offloaded, the number of CPU cycles required to process these data and the latency QoS constraints of the task to be offloaded.

All nearby UAVs receive the discovery message, broadcasted by the SUAV, and extract the message information.

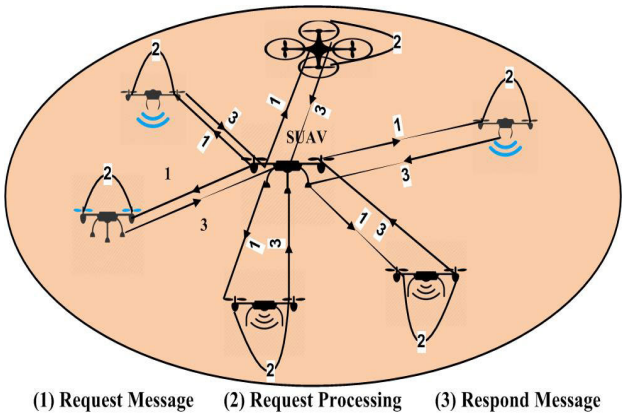

FIGURE 4. Discovery process.

Each UAV decides whether to execute the task and accept the offloading request or to refuse the offloading request, if it cannot handle the task. The decision engine of each nearby UAV calculates the following:

1. Total time required for executing the task defined in the request message T_{ex-UAV}, based on the current available resources,
2. Total energy consumption for processing the task defined in the request message E_{UAV}, based on the current available resources, and
3. Remaining energy of the UAV after processing the target task E_{R-UAV}.

$$T_{ex-UAV} = \frac{N_{CYC}}{R_{UAV}}, \quad R_{UAV} \in f_{UAV} \tag{7}$$

$$E_{UAV} = N_{CYC}\delta_{UAV} \tag{8}$$

$$E_{R-UAV} = E_{C-UAV} - E_{UAV} \tag{9}$$

The decision engine of each UAV calculates the energy decision variable for accepting offloading $D_{E-acc-off}$ by comparing remaining energy after task execution E_{R-UAV} with the threshold energy level of the UAV. If the remaining energy is less than the threshold energy level, the decision engine refuses the offloading request and the energy decision variable of accepting offloading $D_{E-acc-off}$ is set to zero. Else, the UAV accepts the offloading and sends a respond message with the decision to the SUAV. The SUAV has to check T_{UAV} with the QoS latency for positive responses.

$$D_{E-acc-off} = I\left(E_{R-UAV}, E_{th}\right) = \begin{cases} 0 & IF\ (E_{R-UAV} \le E_{th}) \\ 1 & IF\ (E_{R-UAV} > E_{th}) \end{cases} \tag{10}$$

The response message contains three main fields, the identification field, the energy decision field and the execution specifications field. Figure 5 illustrates the main fields and sub-fields of the respond message. The energy decision field is a one bit field, which refers to the agreement of offloading – one for agree and zero for reject. The execution specification field is set only if the binary decision field is set to one. This field indicates the main features of the execution process if the task is decided to be offloaded; these features include the total

Request Message			
Identification		Task Specifications	
ID	Location	Type	Size

Respond Message				
Identification		Energy Decision	Execution details	
ID	Location	One/Zero	Execution Time	Energy Consumption

FIGURE 5. Discovery and response messages.

execution time of the task and the total energy consumption for processing the task.

SUAV receives response messages and decides the air offloading based on the decisions in the received messages and the task execution specifications. For all positive responses, SUAV calculates the total latency for handling a task at a nearby UAV (T_{UAV}). This time delay is calculated by adding the execution time delay at the nearby UAV with positive response T_{ex-UAV} and the communication delay (i.e., packet delivery time for uplink transmission of task input data T_{tx} and the time delay for receiving the computed results T_{rx}).

$$T_{UAV} = T_{ex-UAV} + T_{tx} + T_{rx} \tag{11}$$

The communication latency for delivering data packets of task T_{tx} is the sum of the transmission time of task input data t_{trans} and the propagation delay t_{pro} from the SUAV to the targeted UAV.

$$T_{tx} = t_{trans} + T_{pro} \tag{12}$$

$$t_{pro} = \frac{d_{SUAV,UAV}}{C} \tag{13}$$

$$t_{trans} = \frac{Z}{R_b} \tag{14}$$

The achievable bit rate of the uplink transmission can be calculated using Shannon-Hartley formula [33]:

$$R_b = \omega \log_2 \left(1 + \frac{hp}{\sigma}\right) \tag{15}$$

The decision engine of the SUAV then calculates the decision variable for air-offloading $D_{air-off}$ by comparing the total time delay for handling the offloaded task at a nearby UAV (T_{UAV}) with the QoS latency $\bar{C}$. The air-offloading decision is decided if the T_{UAV} is less than the QoS latency $\bar{C}$.

$$D_{air-off} = I\left(T_{UAV}, \tau\right) = \begin{cases} 1 & IF\ (T_{UAV} \leq \tau) \\ 0 & IF\ (T_{UAV} > \tau) \end{cases} \tag{16}$$

If there is more than one nearby UAV with available resources efficient for task execution and a positive decision response for the air offloading, SUAV calculates the T_{UAV} for each nearby UAV with positive response. Then, the decision engine of the SUAV calculates the air-offloading decision for the nearby UAV with the smallest value of T_{UAV}. For negative air-offloading decision, the SUAV decides the ground offloading through the multilevel MEC system.

TABLE 2. Exchanged messages for ground-offloading.

Message	Type	T_X	R_X	Contents
Request Message	I	SUAV	Micro-cloud	Z, K, $\bar{C}$, SUAV Location
Respond Message	I	Micro-cloud	SUAV	Decision, $D_{ground\text{-}offloading}$ Offloading level
Request Message	II	Micro-cloud	Mini-cloud	Z, K, $\bar{C}$
Respond Message	II	Mini-cloud	Micro-cloud	Decision, $D_{ground\text{-}offloading}$

2) GROUND-OFFLOADING PROCESS

SUAV checks the ground-offloading, when both local execution and air-offloading have failed. SUAV sends a request message to the corresponding micro-cloud edge server connected to the ground base station. The requested message is of type *I*, with the information of the task specifications and vehicle identifications as introduced in Table 2. The micro-cloud edge server receives the offloading request and starts to process the request. The decision engine of the micro-cloud server calculates the total time required to execute the target task based on the current available resources $T_{ex-Micro}$.

$$T_{ex-Micro} = \frac{N_{CYC}}{R_{Micro}}, \quad R_{Micro} \in f_{Micro} \tag{17}$$

The decision engine of the micro-cloud unit then, calculates the total time required to handle the task T_{Micro}.

$$T_{Micro} = T_{ex-Micro} + T_{tx} + T_{rx} \tag{18}$$

$$T_{tx} = t_{trans} + T_{pro} \tag{19}$$

$$t_{pro} = \frac{d_{SUAV,Micro-cloud}}{C} \tag{20}$$

The micro-cloud server calculates the time decision variable for accepting ground-offloading, by comparing total time for handling the task at the micro-cloud server T_{Micro} with the latency QoS time $\bar{C}$.

$$D_{T-Micro-acc-off} = I\left(T_{Micro}, \tau\right) = \begin{cases} 1 & IF\ (T_{Micro} \leq \tau) \\ 0 & IF\ (T_{Micro} > \tau) \end{cases} \tag{21}$$

The positive time decision of accepting task offloading is decided, if the handling time T_{Micro} is less than or at worst equal to the latency QoS time $\bar{C}$. In this case, the decision engine of micro-cloud server checks the energy decision. Otherwise, the micro-cloud server forwards the offloading task to the mini-cloud edge server.

To check the availability of energy resources of micro-cloud server, the decision engine first calculates the total energy consumption to handle the task E_{Micro}. Then, the remaining energy of micro-cloud server $E_{R-Micro}$, if the task is offloaded and handled, is calculated.

$$E_{Micro} = N_{CYC}\delta_{Micro} + T_{trans}P\eta_C \tag{22}$$

$$E_{R-Micro} = E_{C-Micro} - E_{Micro} \tag{23}$$

The decision engine calculates the energy decision variable of accepting ground offloading of the task $D_{E-Micro-acc-off}$ by comparing remaining energy of micro-cloud server $E_{R-Micro}$ with the threshold of energy of micro-cloud server, at which the energy of micro-cloud server is set to be at the critical level.

$$\begin{aligned} D_{E-Micro-acc-off} &= I\left(E_{R-Micro}, E_{th-Micro}\right) \\ &= \begin{cases} 0 & IF\ \left(E_{R-Micro} \leq E_{th-Micro}\right) \\ 1 & IF\ \left(E_{R-Micro} > E_{th-Micro}\right) \end{cases} \end{aligned} \tag{24}$$

For a positive energy decision, the micro-cloud server sends a response message to the SUAV with the agreement of ground-offloading. If the energy decision of accepting offloading is negative, the micro-cloud server sends a request message of type *II* to the corresponding mini-cloud edge server. The mini-cloud edge server receives the request message and process it, by calculating the time and energy decision variables of accepting offloading ($D_{T-Mini-acc-off}$ and $D_{E-Mini-acc-off}$).

$$T_{ex-Mini} = \frac{N_{CYC}}{R_{Mini}}, \quad R_{Mini} \in f_{Mini} \tag{25}$$

$$T_{Mini} = T_{ex-Mini} + T_{tx} + T_{rx} \tag{26}$$

$$\begin{aligned} D_{T-Mini-acc-off} &= I\left(T_{Mini}, \tau\right) \\ &= \begin{cases} 1 & IF\ \left(T_{Mini} \leq \tau\right) \\ 0 & IF\ \left(T_{Mini} > \tau\right) \end{cases} \end{aligned} \tag{27}$$

$$E_{Mini} = N_{CYC}\delta_{Mini} + T_{trans}P_S\eta_C \tag{28}$$

$$E_{R-Mini} = E_{C-Mini} - E_{Mini} \tag{29}$$

$$\begin{aligned} D_{E-Mini-acc-off} &= I\left(E_{R-Mini}, E_{th-Mini}\right) \\ &= \begin{cases} 0 & IF\ \left(E_{R-Mini} \leq E_{th-Mini}\right) \\ 1 & IF\ \left(E_{R-Mini} > E_{th-Mini}\right) \end{cases} \end{aligned} \tag{30}$$

For only positive time and energy decisions, the decision engine of mini-cloud edge server accepts the offloading. Otherwise, the mini-cloud edge server decides to reject the offloading request. The decision engine of mini-cloud unit sends a response message with the decision to the micro-cloud unit. If both micro- and mini-cloud units cannot accept the offloading request, the SUAV block the task.

IV. PERFORMANCE EVALUATION

In this part, the performance of the proposed latency- and energy-aware offloading algorithm is evaluated for UAVs network with heterogeneous capabilities. The ground stations are connected to a multilevel MEC system that provides the path for ground offloading.

A. SIMULATION SETUP

The proposed algorithm is simulated using Matlab, for a system that consists of an air tire and ground tire. The air tire has five drones with heterogeneous processing and energy capabilities. The five drones are assumed to be randomly distributed over the air field of radius R_{air} and altitude h within the range indicated in Table 3. The flying trajectory introduced in [34] is used for UAVs of air tire. The ground tire consists of two cellular cells, with radius R_{cell}, each cell has a base station located at the center of the cell. The base station of the cell A is connected to a micro-cloud edge server with the computing capabilities presented in Table 3. The base station of the neighbor cell B is connected to a mini-cloud edge server with the computing capabilities introduced in Table 3, and both edge servers are connected. The topology considered for the performance evaluation is illustrated in Figure 6. The simulation parameters are summarized in Table 3.

TABLE 3. Simulation parameters.

parameter	value
h	ϵ [50,100] m
R_{air}	2Km
R_{cell}	1Km
δ_{UAV}	1J/GHz
δ_{Micro}	1J/GHz
F_{UAV} (ID=1)	ϵ [0.5,1.5] GHz
F_{UAV} (ID=2)	ϵ [1.0,2.0] GHz
F_{UAV} (ID=3)	ϵ [0.5,1.0] GHz
F_{UAV} (ID=4)	ϵ [0.7,1.2] GHz
F_{UAV} (ID=5)	ϵ [0.5,1.5] GHz
ω	20 MHz
σ	-10 dBm
P	20 dBm
$f_{Micro-cloud}$	ϵ [2.0,4.0] GHz
$f_{Mini-cloud}$	ϵ [3.0,6.0] GHz

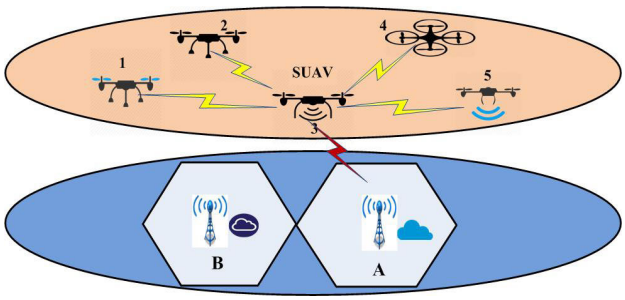

FIGURE 6. Network topology considered for simulation.

Three main scenarios are considered for the simulation purpose, with various considered cases for each scenario.

1) SCENARIO (A)

Ten heterogeneous tasks with different workloads are considered at the SUAV and these tasks come sequentially. The SUAV, based on the proposed offloading algorithm, executes these tasks; either locally or by offloading them to an air- or ground-device based on the available resources. Table 4 indicates the size of the considered tasks, which corresponds to workloads of ten images with different resolutions. The SUAV is assumed to be the UAV with the ID 3. For this scenario, three cases are considered, in each case a certain value of maximum allowable latency that meets the QoS is assumed for each task. Table 5 indicates the values of the QoS in terms of latency for each task in each case.

TABLE 4. Task specifications.

Task	Task(1)	Task(2)	Task(3)	Task(4)	Task(5)
Z(kB)	200	270	320	250	370
Task	Task(6)	Task(7)	Task(8)	Task(9)	Task(10)
Z(kB)	400	450	390	500	520

TABLE 5. QoS latency for different cases.

Task	Task(1)	Task(2)	Task(3)	Task(4)	Task(5)
C_1(ms)	10	13.5	16	12.5	18.5
C_2(ms)	15	20.25	24	18.75	27.75
C_3(ms)	20	27	32	25	37
Task	Task(6)	Task(7)	Task(8)	Task(9)	Task(10)
C_1(ms)	20	22.5	19.5	25	26
C_2(ms)	30	33.75	29.25	37.5	39
C_3(ms)	40	45	39	50	52

TABLE 6. Allocated tasks for each drone.

Drone ID	Drone 1	Drone 2	Drone 3	Drone 4	Drone 5
Allocated Task	3,7	1,6,9	2,4	10	5,8

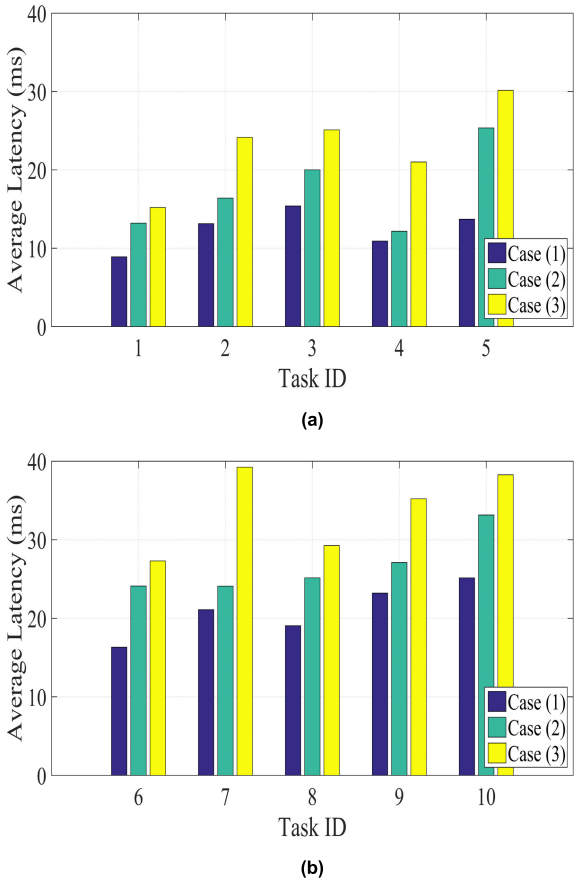

FIGURE 7. Average latency of tasks handling for different cases of scenario (A).

2) SCENARIO (B)

In this scenario, the previously considered tasks are randomly distributed among the five drones. Table 6 indicates the allocated tasks for each UAV. Tasks are executed simultaneously, however UAV with more than one task are queued until the current task is executed. The three considered cases in the previous scenario are also considered in this scenario.

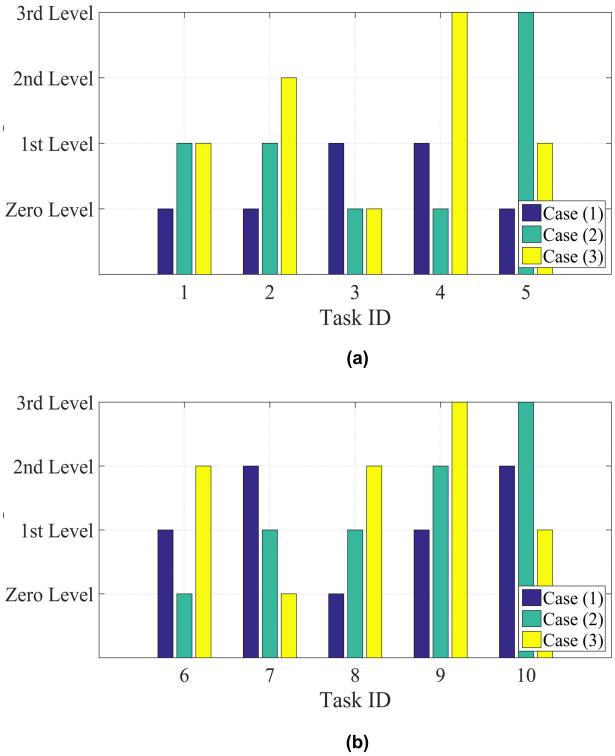

FIGURE 8. Offloading levels of each task for different cases of scenario (A).

3) SCENARIO (C)

This scenario is proposed for evaluating the performance and importance of each offloading level. In this scenario, the system is simulated five times; each time represents a simulation case. For the first case, the system is simulated without offloading. In the second case, the system is simulated with considering only air-offloading and it is assumed that there is no possibility for ground-offloading. The third case, assumes the existence of ground-offloading only and there is no possibility for the air-offloading. The ground-offloading in this case is assumed to be homogenous MEC servers (i.e., micro-cloud units) and there is no existence of mini-cloud units. In the fourth case, the ground offloading is assumed only with no air-offloading, while the micro-cloud and mini-cloud units are both deployed. In the last case, the system is simulated with the deployment of both; air- and ground-offloading with multilevel MEC. The considered tasks are the same as in the first scenario with the SUAV with the ID 3. Thus, scenario (A) can be considered to be the fifth case of scenario (c). Simulation scenario (c) is considered for indicating the importance and impact of each of the considered offloading level on the performance.

B. SIMULATION RESULTS AND ANALYSIS

Figure 7 illustrates the average latency for handling each task at each case of the simulation scenario (A). Some considered tasks are handled locally at the SUAV with no need

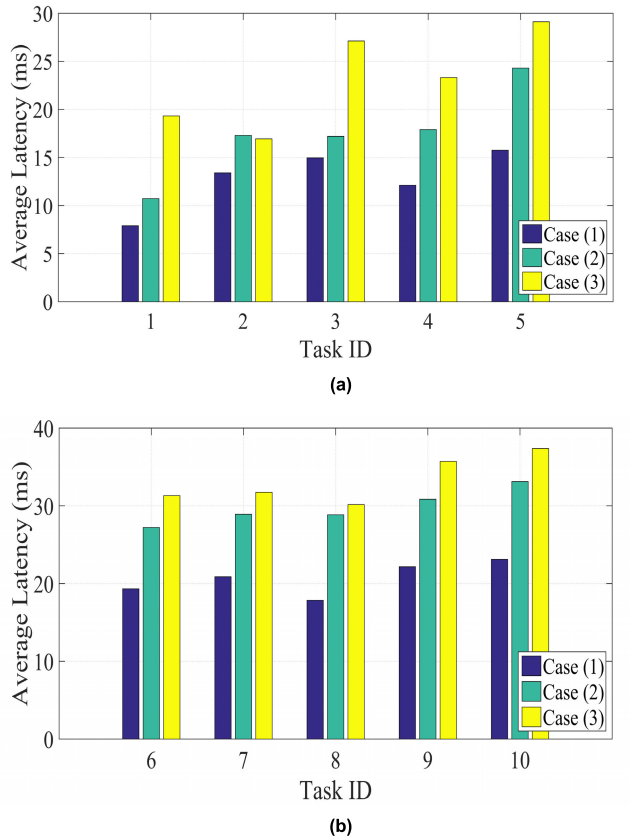

FIGURE 9. Average latency of tasks handling for different cases of scenario (B).

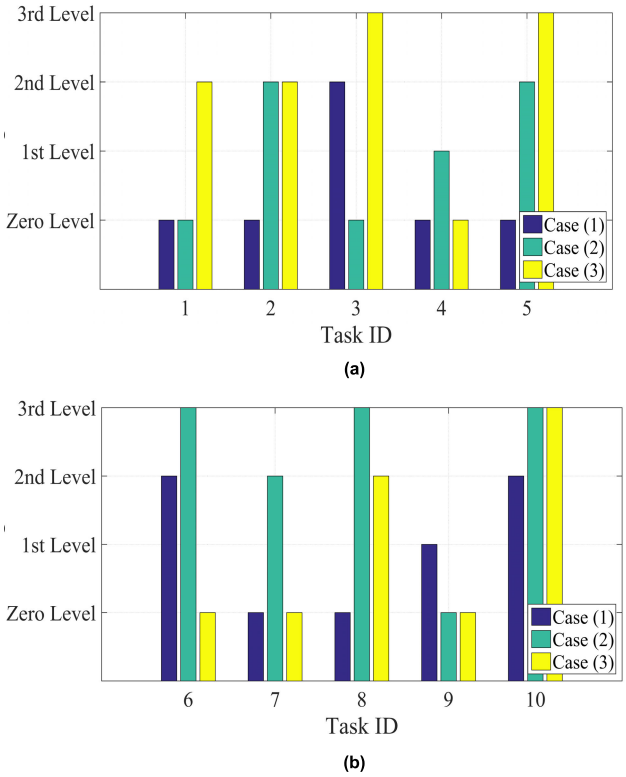

FIGURE 10. Offloading levels of each task for different cases of scenario (B).

for offloading, while other tasks are offloaded because of the available resources are not sufficient for task handling within the latency requirement. Figure 8 indicates the level of offloading for each task in each case. A number of the offloaded tasks are executed at nearby UAVs, while the rest are offloaded to the ground multilevel MEC servers where these tasks are executed. With no ground offloading a number of SUAV tasks could not be handled and therefore some tasks would be blocked.

Results for simulation scenario (B) are presented in Figures 9 and 10. Figure 9 illustrates the average latency for handling each task at each considered case of scenario (B), while Figure 10 presents the offloading level for each task at each case, which indicates where the each task is executed. For this scenario, all UAVs are assigned tasks and this reduces the probability that a nearby UAV has available resources to support air-offloading. Tasks 1 – 5 and 10 are allocated for the five drones at the same time, thus each UAV has a task to execute. Therefore, these tasks are either executed locally (i.e., zero offloading level) or ground-offloaded (i.e., second offloading or third offloading levels).

For evaluating the importance and effectiveness of each offloading level, the simulation scenario (C) is considered. Figure 11 illustrates the comparison of the five considered cases of simulation scenario (C) in terms of the percentage of blocked tasks. The system is first deployed with no offloading and tasks are handled sequentially. Some tasks are handled locally and other tasks are blocked due to the unavailability of resources (e.g., the available resources handle the task in a time greater than latency QoS time $\bar{C}$). The blocked tasks were recorded and the percentage of blocked tasks is indicated in Figure 11. This is repeated three times, each time corresponds to a value of threshold latency $\bar{C}$ indicated in Table 5. For case (2), the system is run with the local execution and the air-offloading, and the total number of blocked tasks was recorded. As indicated in Figure 11, the number of blocked tasks is reduced with the deployment of air-offloading besides the local processing. The third case, consider the ground offloading besides the local offloading and assume no air-offloading. In this case, the number of blocked tasks is reduced compared to the previous two cases, but there is still some blocked tasks. This implies that the ground offloading besides the local processing is not sufficient and also there is still blocking. Thus, the UAVs needs both offloading; air-offloading and ground-offloading to achieve the best performance. This represents the fifth case, which indicates zero blocking as illustrated in Figure 11.

In order to present the value of each offloading level, and the impact of adding these levels on the performance, the percentage improvement of task blocking is calculated for each of offloading level with respect to the local execution. Table 7 indicates the percentage improvement of task blocking achieved by each offloading level with respect to local processing for the previous considered scenario for three considered values of the QoS latency. Furthermore, the average value of the percentage improvement for the three

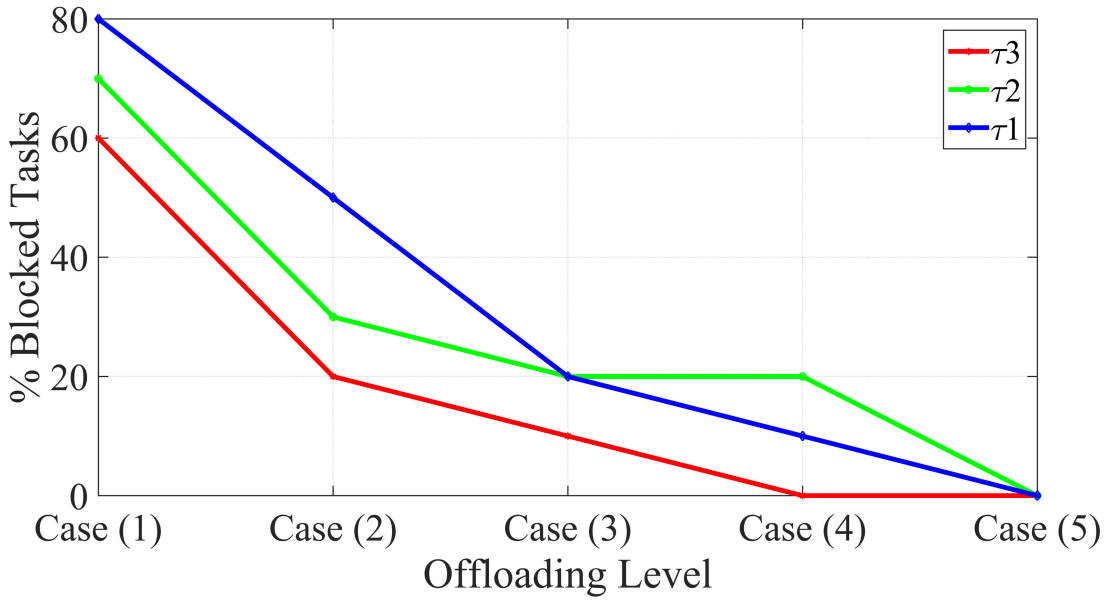

FIGURE 11. Percentage of blocked tasks for each case of scenario (C).

TABLE 7. Percentage improvement of task blocking for each offloading level.

Offloading level	First level offloading	Second level offloading	2nd and 3rd levels offloading	All offloading levels
	Air-offloading	Micro-cloud offloading	Ground-offloading	Air & Ground offloading
C_3	67%	83.3%	100%	100%
C_2	57%	71.4%	71.4%	100%
C_1	37%	75%	87.5%	100%
Average	53.8%	76.6%	86.3%	100%

considered cases of latency is presented. Results indicate that employing both ground and air offloading achieves the complete improvement of the system performance in terms of blocked tasks.

V. CONCLUSION

This work presented an energy- and latency-aware offloading algorithm for UAVs for time-critical applications. The algorithm comprises three main routines; the first is responsible for deciding the location of task execution or the offloading method if required. The decision is made by a decision engine, which decides based on the available resources and the required QoS constraints. The second routine is responsible for the air-offloading process, which enables SUAV to offload their computing tasks to nearby UAVs with available computing and energy resources. The third routine is responsible for ground offloading, which is deployed if the air-offloading fails. The ground offloading is based on a multilevel MEC system that provides heterogeneous computing capabilities at the edge of the RAN. The proposed algorithm achieves higher efficiency in terms of latency and blocking probability.

In future work, we intend to address the problem of offloading to air and ground devices, e.g., Internet of Thing (IoT)-enabled spaces [35] or city infrastructure [36]. Another avenue of research is to investigate the security of the UAV network especially in regards to authentication. Many lightweight authentication techniques that are designed specifically for IoT, e.g., [37] can theoretically be applied to UAV networks.

REFERENCES

[1] *Feasibility Study on New Services and Markets Technology Enablers*, document 3GPP TR 22.891, ver. 14.2.0, Sep. 2016.
[2] O. Aldabbas, A. Abuarqoub, M. Hammoudeh, U. Raza, and A. Bounceur, "Unmanned ground vehicle for data collection in wireless sensor networks: Mobility-aware sink selection," *Open Autom. Control Syst. J.*, vol. 8, no. 1, Jul. 2016.
[3] N. H. Motlagh, M. Bagaa, and T. Taleb, "UAV-based IoT platform: A crowd surveillance use case," *IEEE Commun. Mag.*, vol. 55, no. 2, pp. 128–134, Feb. 2017.
[4] S. A. A. Shah, E. Ahmed, M. Imran, and S. Zeadally, "5G for vehicular communications," *IEEE Commun. Mag.*, vol. 56, no. 1, pp. 111–117, Jan. 2018.
[5] H. Shakhatreh *et al.* (2018). "Unmanned aerial vehicles: A survey on civil applications and key research challenges." [Online]. Available: https://arxiv.org/abs/1805.00881
[6] S. Sekander, H. Tabassum, and E. Hossain, "Multi-tier drone architecture for 5G/B5G cellular networks: Challenges, trends, and prospects," *IEEE Commun. Mag.*, vol. 56, no. 3, pp. 96–103, Mar. 2018.
[7] T. Zahariadis, A. Voulkidis, P. Karkazis, and P. Trakadas, "Preventive maintenance of critical infrastructures using 5G networks drones," in *Proc. 14th IEEE Int. Conf. Adv. Video Signal Based Surveill. (AVSS)*, Sep. 2017, pp. 1–4.
[8] Y. Zeng, J. Lyu, and R. Zhang. (2018). "Cellular-connected UAV: Potentials, challenges and promising technologies." [Online]. Available: https://arxiv.org/abs/1804.02217

[9] A. Thibbotuwawa, P. Nielsen, B. Zbigniew, and G. Bocewicz, "Energy consumption in unmanned aerial vehicles: A review of energy consumption models and their relation to the UAV routing," in *Proc. 39th Int. Conf. Inf. Syst. Archit. Technol.* Cham, Switzerland: Springer, 2018, pp. 173–184.
[10] E. Ahmed *et al.*, "Bringing computation closer toward the user network: Is edge computing the solution?" *IEEE Commun. Mag.*, vol. 55, no. 11, pp. 138–144, Nov. 2017.
[11] W. Khawaja, I. Guvenc, D. Matolak, U. C. Fiebig, and N. Schneckenberger. (2018). "A survey of air-to-ground propagation channel modeling for unmanned aerial vehicles." [Online]. Available: https://arxiv.org/abs/1801.01656
[12] R. Valentino, W. S. Jung, and Y. B. Ko, "Opportunistic computational offloading system for clusters of drones," in *Proc. 20th Int. Conf. Adv. Commun. Technol.*, Feb. 2018, pp. 303–306.
[13] F. Cheng *et al.*, "UAV trajectory optimization for data offloading at the edge of multiple cells," *IEEE Trans. Veh. Technol.*, vol. 67, no. 7, pp. 6732–6736, Jul. 2018.
[14] *Management and Orchestration; 5G end to end Key Performance Indicators (KPI), Release 15*, document 3GPP TS 28.554, ver. 2.0.0, Sep. 2018.
[15] E. Ahmed, A. Naveed, A. Gani, S. H. A. Hamid, M. Imran, and M. Guizani, "Process state synchronization-based application execution management for mobile edge/cloud computing," *Future Gener. Comput. Syst.*, vol. 91, no. 2, pp. 579–589, Feb. 2019.
[16] B. Kim, H. Min, J. Heo, and J. Jung, "Dynamic computation offloading scheme for drone-based surveillance systems," *Sensors*, vol. 18, no. 9, p. 2982, 2018.
[17] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing a key technology towards 5G," ETSI, Sophia Antipolis, France, White Paper 11, Sep. 2015.
[18] M. A. Messous, H. Sedjelmaci, N. Houari, and S. M. Senouci, "Computation offloading game for an UAV network in mobile edge computing," in *Proc. IEEE Int. Conf. Commun.*, May 2017, pp. 1–6.
[19] M. Narang *et al.*, "UAV-assisted edge infrastructure for challenged networks," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, May 2017, pp. 60–65.
[20] N. Hassan, S. Gillani, E. Ahmed, I. Ibrar, and M. Imran, "The role of edge computing in Internet of Things," *IEEE Commun. Mag.*, vol. 56, no. 11, pp. 110–115, Nov. 2018. doi: 10.1109/MCOM.2018.1700906.
[21] Z. Zhao *et al.*, "Software-defined unmanned aerial vehicles networking for video dissemination services," *Ad Hoc Netw.*, vol. 83, pp. 68–77, Feb. 2019.
[22] A. A. Ateya, A. Muthanna, A. Vybornova, P. Darya, and A. Koucheryavy, "Energy-aware offloading algorithm for multi-level cloud based 5G system," in *Proc. Internet of Things, Smart Spaces, Next Gener. Netw. Syst.* Cham, Switzerland: Springer, Aug. 2018, pp. 355–370.
[23] A. A. Ateya, A. Vybornova, R. Kirichek, and A. Koucheryavy, "Multilevel cloud based Tactile Internet system," in *Proc. IEEE 19th Int. Conf. Adv. Commun. Technol. (ICACT)*, Bongpyeong, South Korea, Feb. 2017, pp. 105–110.
[24] C. Luo, J. Nightingale, E. Asemota, and C. Grecos, "A UAV-cloud system for disaster sensing applications," in *Proc. IEEE 81st Veh. Technol. Conf. (VTC Spring)*, May 2015, pp. 1–5.
[25] J. Lynskey, "Maximizing offloading opportunities for UAV communication," Korean Netw. Oper. Manage., Jeju Nat. Univ., Jeju, South Korea, Tech. Rep., May 2018.
[26] Z. Zhou, J. Feng, L. Tan, Y. He, and J. Gong, "An air-ground integration approach for mobile edge computing in IoT," *IEEE Commun. Mag.*, vol. 56, no. 8, pp. 40–47, Aug. 2018.
[27] W. S. Jung, J. Yim, and Y. B. Ko, "Adaptive offloading with MPTCP for unmanned aerial vehicle surveillance system," *Ann. Telecommun.*, vol. 73, pp. 1–14, Oct. 2018.
[28] R. M. Shukla, S. Sengupta, and A. N. Patra, "Software-defined network based resource allocation in distributed servers for unmanned aerial vehicles," in *Proc. IEEE 8th Annu. Comput. Commun. Workshop Conf. (CCWC)*, Jan. 2018, pp. 796–802.
[29] A. A. Ateya, A. Vybornova, K. Samouylov, and A. Koucheryavy, "System model for multi-level cloud based tactile Internet system," in *Proc. Int. Conf. Wired/Wireless Internet Commun.*, Mar. 2017, pp. 77–86.
[30] V. Sharma, I. You, G. Pau, M. Collotta, J. D. Lim, and J. N. Kim, "LoRaWAN-based energy-efficient surveillance by drones for intelligent transportation systems," *Energies*, vol. 11, no. 3, p. 573, 2018.
[31] L. Krichen, M. Fourati, and L. C. Fourati, "Communication architecture for unmanned aerial vehicle system," in *Proc. Int. Conf. Ad-Hoc Netw. Wireless*. Cham, Switzerland: Springer, Sep. 2018, pp. 213–225.
[32] G. Secinti, P. B. Darian, B. Canberk, and K. R. Chowdhury, "SDNs in the Sky: Robust end-to-end connectivity for aerial vehicular networks," *IEEE Commun. Mag.*, vol. 56, no. 1, pp. 16–21, Jan. 2018.
[33] A. Kartun-Giles, S. Jayaprakasam, and S. Kim, "Euclidean matchings in ultra-dense networks,"*IEEE Commun. Lett.* vol. 22, no. 6, pp. 1216–1219, Jun. 2018.
[34] V. Sharma, M. Bennis, and R. Kumar, "UAV-assisted heterogeneous networks for capacity enhancement," *IEEE Commun. Lett.*, vol. 20, no. 6, pp. 1207–1210, Jun. 2016.
[35] A. Coates, M. Hammoudeh, and K. G. Holmes, "Internet of things for buildings monitoring: Experiences and challenges," in *Proc. Int. Conf. Future Netw. Distrib. Syst.*., ACM, Jul. 2017, p. 38.
[36] O. Jogunola *et al.*, "Distributed adaptive primal algorithm for P2P-ETS over unreliable communication links," *Energies*, vol. 11, no. 9, p. 2331, 2018.
[37] Y. Atwady and M. Hammoudeh, "A survey on authentication techniques for the Internet of Things," in *Proc. Int. Conf. Future Netw. Distrib. Syst.*, Jul. 2017, p. 8.

ABDELHAMIED A. ATEYA (M'18) received the B.Sc. and M.Sc. degrees in electrical engineering from Zagazig University, Egypt, in 2010 and 2014, respectively, where he is currently an Assistant Lecturer with the Faculty of Engineering. He is currently pursuing the Ph.D. degree with St. Petersburg State University of Telecommunication, St. Petersburg, Russia. His main research area includes wireless communications. He is currently working on 5G systems and future Tactile Internet.

AMMAR MUTHANNA received the B.Sc., M.Sc., and Ph.D. degrees from St. Petersburg State University of Telecommunications, Russia, in 2009, 2011, and 2016, respectively. In 2012 and 2013, he took part in the Erasmus Student Program with the Faculty of Electrical Engineering, University of Ljubljana. He is currently an Associate Professor with the Department of Telecommunication Networks, and also the Head of the SDN Laboratory (sdnlab.ru), St. Petersburg State University of Telecommunications. His research areas include wireless communications, 4G/5G cellular systems, the IoT applications, and software-defined networking.

RUSLAN KIRICHEK was born in Tartu, Estonia, 1982. He received the degrees from the Military-Space Academy A.F. Mozhaiskogo and The Bonch-Bruevich University of Telecommunication, St. Petersburg, Russia, in 2004 and 2007, respectively, and the Ph.D. degree from St. Petersburg University of Telecommunication, in 2012.

Since 2004, he has been a Senior Engineer with the IT-Department of the Air Force. Since 2008, he has been a Senior Researcher with the Federal State Unitary Enterprise Center-Inform, where he has supervised research testing communication networks in terms of destructive influences. Since 2012, he has been the Head of the Internet of Things Laboratory, State University of Telecommunication. He is currently an Associate Professor with the Department of Communications Networks, The Bonch-Bruevich University of Telecommunication.

MOHAMMAD HAMMOUDEH has been a Researcher, and also a Publisher in the field of big sensory data mining and visualization. He is currently the Head of the MMU IoT Laboratory, and a Senior Lecturer in computer networks and security with the School of Computing, Math and Digital Technology, Manchester Metropolitan University. He is the External Examiner for the B.Sc. (Hons.) computer security and forensics with the University of Bedfordshire. He is also the External Examiner for the M.Sc. computer networks and security and M.Sc. cloud computing awards at the Staffordshire University. He is the HEA Institutional Representative (information and computer sciences subject). He is a highly proficient, experienced, and professionally certified cybersecurity professional, specialized in threat analysis and information and network security management. His research interests include highly decentralized algorithms, communication, and cross-layered solutions to the Internet of Things and wireless sensor networks.

Mr. Hammoudeh is a Senior Member of the IEEE. He has been an Active Member of the Technical Program Committee on many international conferences and journals. He is regularly invited to talk at the international conferences and workshops.

ANDREY KOUCHERYAVY was born in Leningrad, USSR, in 1952. He received the Ph.D. and Dr.Sc. degrees, in 1982 and 1994, respectively. After graduation from the Leningrad University of Telecommunication, in 1974, he joined the Telecommunication Research Institute LONIIS, where he was the first Deputy Director, from 1986 to 2003.

Since 1998, he has been a Professor with St. Petersburg State University of Telecommunication (SUT). He has been a Chaired Professor with the Department of Telecommunication Networks and Data Transmission, since 2011. His scientific areas of interest are network planning, teletraffic theory, and the IoT and its enablers.

Dr. Koucheryavy is a honorary member of the A.S. Popov's Society. He is the Chairman of the Study Group 11 ITU-T (Study periods 2017–2020).