Energy-Efficient Task Offloading and Trajectory Design for UAV-based MEC Systems

Mohamed El-Emary, Ali Ranjha, Diala Naboulsi, Razvan Stanica

Energy-Efficient Task Offloading and Trajectory Design for UAV-based MEC Systems

Mohamed El-Emary*, Ali Ranjha*, Diala Naboulsi*, and Razvan Stanica†
*Department of Software Engineering and IT, École de Technologie Supérieure, Montréal, QC, H3C 1K3, Canada
†Univ Lyon, INSA Lyon, Inria, CITI, F-69621 Villeurbanne, France

***Abstract*—Sixth-generation and mobile edge computing (MEC) systems are expected to empower a wide range of applications. Unmanned aerial vehicles (UAVs) can play a vital role in improving network connectivity. Hence, our problem is to minimize the user equipment (UE) energy consumption during task offloading in a UAV assisted MEC system. To address the formulated NP-hard problem, we propose task scheduling and assignment algorithms for mapping UE tasks to fixed edge servers using UAV. Lastly, the simulation results demonstrate that the proposed algorithms yield better results than other benchmark methods in terms of total UE energy consumption.**

***Index Terms*—unmanned aerial vehicle (UAV), mobile edge computing (MEC), task scheduling, trajectory design.**

I. Introduction

Upcoming sixth-generation (6G) mobile networks are expected to give rise to a slew of new mobile applications, including mobile online gaming, augmented reality, and smart navigation, among others [1]–[3]. Due to low battery capacity, user equipment (UE) is unable to perform all computational tasks locally at their end. To remedy this problem, a UE can offload resource-intensive computational work to a nearby server that can perform the tasks on the UE behalf, significantly increasing the UE capabilities by reducing its energy consumption. In such a case, the UE can transmit its computation tasks to the nearest base station (BS) hosting a mobile edge computing (MEC) server and wait for the result [4].

Leveraging an unmanned aerial vehicle (UAV)-assisted MEC system is envisioned to be a feasible solution in case there are no available reachable servers [5]. This solution is crucial for providing communication at places with inadequate infrastructure [6]. In this regard, some challenges arise concerning task offloading in UAV-assisted MEC systems, including UAV deployment, device association between UE and UAVs, as well as task scheduling [7]. Addressing these challenges can help guarantee UE quality-of-service (QoS) during task offloading and lower its energy consumption by matching tasks with computing resources [8]. Furthermore, task offloading is constrained by UE mobility, resulting in processing tasks with non-uniform arrival, and network scalability issues, produced by numerous UEs simultaneously offloading tasks. Therefore, efficient UAV deployment is critical to allow UEs to reach fixed servers at the BSs to offload their tasks and reduce their energy consumption [9].

Recently, the problem of task offloading in UAV-based MEC systems has gained a lot of attention from the research community. Yet, researchers have focused more in their studies on the UAV performance than on the UE performance. Therefore, in our work, we study the problem, with the objective of minimizing the UE energy consumption. We prove this problem to be NP-hard. This inspired us to implement an earliest deadline first (EDF) algorithm to prioritize UE tasks based on their deadline, and a meta-heuristic differential evolution algorithm (DEA) to plan the UAV trajectory, providing UEs with feasible routes to unreachable BSs (and servers). Our results show that the proposed algorithms eventually reduce the UE energy consumption and yield better performance when compared to non-UAV-assisted benchmark methods.

The rest of this paper is organized as follows. In Section II, we present the related work and the literature review. Section III presents the system model and the problem formulation. Section IV describes our proposed algorithms. In Section V, we evaluate the performance of the proposed algorithms. Finally, in Section VI, we conclude the paper.

II. Related Work

As mentioned above, only a few researchers focused on the performance on the UE side, with contributions aimed at minimizing the end to end task execution delay [10], [11]. In [10], the authors studied the minimization problem of the total system delay including flying delay, transmission delay, UE local computing delay and UAV-aided edge computing delay through joint optimization of the flying UAV trajectory and the ratio of the offloading tasks using UAV passive fault tolerant control. To solve this problem, a machine learning framework based on Q-Learning algorithm is proposed to minimize the total delay of the system. Similarly, in [11], the authors sought to execute the offloaded tasks before their deadline while simultaneously considering the UAV hover time for a passive fault tolerant control. To achieve this, the authors introduced a multi objective maximization problem and a Q-Learning based method.

While these works [10]–[12] focused on the delay and the response time on UE side, other targeted minimizing the energy consumption of both the UEs and UAVs in a multi UAV-assisted framework. In [13], authors proposed ENergy-efficient disastER manaGmENT (ENERGENT) as a novel framework for disaster management in the UAV-assisted Internet of Things (IoT) Fog networks. ENERGENT optimizes the energy consumption of the UEs, as well as the UAVs, using three proposed algorithms.The first algorithm adjusts the 3D placement of the UAVs such that these nodes consume the minimum energy to reach the desired cluster of the TNs. Besides, the transmit power and the transmission rate of the UEs are set in a way that their energy consumption is minimized and the outage probability requirements are met in the network. In the second algorithm, a task offloading scheme is proposed where tasks are offloaded to the UAVs in order to meet the network delay constraints. Finally, the third algorithm takes advantage of wireless power transfer to transfer energy to the UEs when their remaining energy degrades below a predefined threshold.

Moreover, in [14], a UAV-enabled Computing-Communications Intelligent Offloading scheme is proposed to offload tasks energy-efficiently. First, some nodes with a large amount of data (i.e. numerous tasks) are selected as Task Gathering Nodes (TGNs). TGNs further collect all the tasks of the remaining nodes. In this way, the UAV can gather all the IoT devices' tasks through the TGNs. The travel distance needed for the UAV can be greatly reduced and both UAV and IoT energies are saved. On the other hand, tasks that are routed to TGNs have a relatively small amount of data, while nodes with a large amount of data have already been selected as TGNs without routing. Second, an optimization strategy for collection tasks is proposed to reduce UAV energy consumption.

Similarly, in [15], digital twin (DT) technology is used to map the physical entities to virtual models, and reflect the MEC network state in real-time. In this paper, authors proposed a MEC network with multiple movable UAVs and one DT-empowered ground base station to enhance the MEC service for the UEs. Considering the limited energy resource of both UEs and UAVs, an online problem of resource scheduling is formulated to minimize their total weighted energy consumption. To solve the resulting combinatorial problem, a Markov decision process (MDP) modeling is considered, with multiple types of agents. Since the proposed MDP has large state and action spaces, a deep reinforcement learning approach is proposed, based on multi-agent proximal policy optimization with Beta distribution and attention mechanism to pursue the optimal computation offloading policy.

Furthermore, in [16], a novel data offloading decision-making framework is proposed, where users have the option to partially offload their data to a complex MEC environment, consisting of both ground and UAV-mounted MEC servers. The problem is treated under the perspective of risk-aware user behavior as captured via prospect-theoretic utility functions, while accounting for the inherent computing environment uncertainties. The UAV-mounted MEC servers act as a common pool of resources with potentially superior, but uncertain, payoff for the users, while the local computation and ground server alternatives constitute safe and guaranteed options. The optimal user task offloading to the available computing choices is formulated as a maximization problem of each user's satisfaction, and confronted as a non-cooperative game. The existence and uniqueness of a Pure Nash Equilibrium (PNE) are proven, and convergence to the PNE is shown.

Overall, we observe that none of the existing works focused solely on the UE energy consumption in a single UAV-assisted MEC system. Therefore, in our paper, we study UE task offloading and plan the UAV trajectory in a UAV-assisted MEC system, with on-ground computing servers, with the objective of minimizing UE energy consumption. The considered problem is NP-hard. In this regard, an EDF algorithm is implemented to sort tasks for offloading based on their deadline. Thereafter, a meta-heuristic DEA is proposed to plan the UAV trajectory and provide UEs with shortest distances to BSs, resulting in a reduction of the UE energy consumption during task offloading. In addition, the UAV reports to the UE any changes in the feasible distances to the BS, e.g. when a BS server is down or busy, yielding better results than non UAV-assisted benchmark methods.

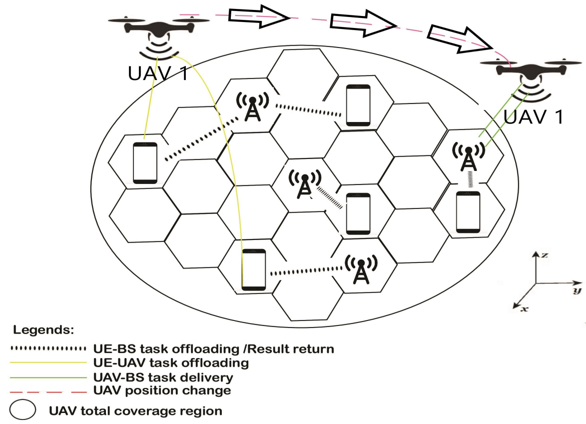

Fig. 1: System Model of the UAV-based MEC system.

III. System Model and Problem Formulation

We consider a set of UE devices in a certain geographical area. We use d to denote one UE device, and D to refer to the set of all UE devices, as shown in Fig. 1. Moreover, we consider a set of BSs is deployed in the area with each one hosting one MEC server. In this regard, we use s to refer to one server and S to refer to all MEC servers. Additionally, we consider a single UAV U is available and can help offload tasks from UE devices to servers. Without loss of generality, we consider each UE device has a set of computational tasks to offload. The tasks can be offloaded directly to nearby MEC servers or to farther ones, with the UAV acting as a relay, in case nearby MEC servers are not available. We use j to denote one computational task and J to refer to the set of all computation tasks. Furthermore, J_d denotes the set of tasks of a device d.

Additionally, the set P contains two types of routes including P_{static}, which refers to the set of static routes, and $P_{dynamic}$, referring to the set of dynamic routes. P_{static} includes routes between UEs and BSs configured in advance of any UAV involvement. $P_{dynamic}$ is the set of routes that result from the deployment of the UAV, i.e. routes that involve the UAV as a relay. The main mathematical notations are summarized in **Table 1**.

TABLE I: Table of Notations

Notation	Definition	Notation	Definition
D	Set of all UE devices	J	Set of all computation tasks
S	Set of MEC servers	U	UAV
J_d	Set of tasks of a device d	P	Set of routes
t_j^{route}	Task j routing time from UE to BS	T	Set of time instants of interest
T_j^{E2E}	End to End delay for task j	T_j^{Exp}	Task j expiry time
T_j^{Arr}	Task j arrival time	E_d	UE d energy consumption
H_{dsp}	Number of hops from UE d to server s	G_0	Positive constant
$t_j^{execute}$	Task j execution time	t_j^{return}	Task j result return time
$E_d^{offload}$	UE energy consumption in case of offloading the task	E_d^{tx}	UE d transmission energy per offloaded task
E_d^{rx}	Total UE d receive energy for its offloaded tasks	E_j^d	The energy consumed for performing task j locally
E_d^{local}	Total UE d energy consumption for all its locally executed tasks	k_d^j	Total number of CPU cycles for task j to be completed
f_d	The processing frequency of UE device d	C_d	The effective switched capacitance of end device d processor
r_{ds}	The uplink data rate between d and s	r_{sd}	The channel downlink data rate
B_d	UE d channel bandwidth	β_0^d	channel power gain at a reference distance 1 m
N_0	Noise power spectrum density	$dist_{ds}$	Euclidean distance between d and s
mut_p	The route to server reported by UAV deployed in a mutated location.	$U(g+1)$	The UAV new mutated location
$POP(X,g)$	The population in the g^{th} generation, where X represents the UAV location	$X(g)$	The x and y coordinates of the UAV obtained in the g^{th} generation.
$P_{Feasible}$	Denotes the best feasible route for set of devices D to offload their tasks J_d		

We assume time is slotted and use t to refer to a single time slot and T to refer to a set of time instants of interest. Accordingly, we want to ensure that the end to end delay T_j^{E2E} of a task j is less than or equal to the task expiry time T_j^{Exp} minus the task arrival time T_j^{Arr}, which is mathematically modelled as:

$$\mathcal{C}_1 : T_j^{E2E} \leq T_j^{Exp} - T_j^{Arr}, \forall j \in J, \tag{1}$$

We limit the UE energy consumption E_d to a minimum E_{min} and maximum E_{max} threshold to maintain the UE battery level. These two constraints are represented as:

$$\mathcal{C}_2 : E_d \leq E_{max}, \forall d \in D, \tag{2}$$

$$\mathcal{C}_3 : E_d \geq E_{min}, \forall d \in D. \tag{3}$$

We define a_p^t as a decision variable that is set to 1 if a path p at time slot t is activated and 0 otherwise. For a dynamic path, we consider that having $a_p^t = 1$ implies that the UAV is positioned adequately so as to make the path available. For static ground paths, we assume they are available at all time by enforcing the activation of a_p^t, through the following equation:

$$\mathcal{C}_4 : a_p^t = 1, \forall p \in P_{static}, \forall t \in T, \tag{4}$$

Furthermore, we define x_{js}^t as the server selection decision variable, which is equal to 1 if task j is executed over server s at time slot t and 0 otherwise. We define accordingly the following constraint to ensure a task j is executed only once over a single server s and over one time slot t:

$$\mathcal{C}_5 : \sum_{s\in S}\sum_{t\in T} x_{js}^t = 1, \forall j \in J, \tag{5}$$

Moreover, we define t_j^{route} as the task routing time from UE to BS. It can be obtained by considering the number of hops H_{dsp} from UE d to server s through path p. We also define y_{jsp}^t as a decision variable that is equal to 1 if task j, executed on server s, will be routed through path p at time t and otherwise 0. t_j^{route} can thus be obtained and controlled by the following equations:

$$t_j^{route} = \sum_{p\in P}\sum_{s\in S}\sum_{t\in T}\sum_{d\in D} y_{jsp}^t \cdot x_{js}^t \cdot H_{dsp} \cdot a_p^{t-1}, \tag{6}$$

$$\mathcal{C}_6 : \sum_{s\in S}\sum_{t\in T}\sum_{p\in P} y_{jsp}^t = 1, \forall j \in J. \tag{7}$$

Now, t_j^{return} is defined as the task result return time. Similarly to t_j^{route}, it can be obtained based on z_{jsp}^t, a decision variable that is equal to 1 if the result of task j, executed on server s, will be routed through path p at time t and otherwise 0. t_j^{return} can thus be obtained and controlled by the following equations:

$$t_j^{return} = \sum_{p\in P}\sum_{s\in S}\sum_{t\in T}\sum_{d\in D} z_{jsp}^t \cdot x_{js}^t \cdot H_{dsp} \cdot a_p^{t+1}. \tag{8}$$

$$\mathcal{C}_7: \sum_{s\in S}\sum_{t\in T}\sum_{p\in P} z^t_{jsp} = 1, \forall j \in J, \tag{9}$$

Thereafter, $t^{execute}_j$ is defined as the task execution time and is obtained based on x^t_{js}. We assume that a task is executed in one time slot by the server, $t^{execute}_j$ is then calculated as:

$$t^{execute}_j = \sum_{p\in P}\sum_{s\in S}\sum_{t\in T}\sum_{d\in D} x^t_{js}, \tag{10}$$

Thus, the end-to-end (E2E) delay is given as :

$$T^{E2E}_j = t^{route}_j + t^{execute}_j + t^{return}_j, \tag{11}$$

Additionally, we define the UE total energy consumption E_d divided into two components: the energy consumption in case of offloading the task to other node $E^{offload}_d$ and the energy consumption in case of computing the task locally on the UE E^{local}_d, which is given as:

$$\begin{aligned}
&E_d = E^{offload}_d + E^{local}_d,\\
&E^{offload}_d = E^{tx}_d + E^{rx}_d,\\
&E^{tx}_d = \sum_{p\in P}\sum_{s\in S}\sum_{j\in J_d}\sum_{t\in T} y^t_{jsp}\cdot E^{transmit}_j,\\
&E^{rx}_d = \sum_{p\in P}\sum_{s\in S}\sum_{j\in J_d}\sum_{t\in T} z^t_{jsp}\cdot E^{receive}_j,\\
&E^{local}_d = \sum_{j\in J}(1-\sum_{t\in T}\sum_{s\in S} x^t_{js})\cdot E^d_j.
\end{aligned} \tag{12}$$

where $E^{offload}_d$ is defined as the summation of transmission energy E^{tx}_d and the receive energy E^{rx}_d. $E^{transmit}_j$ is defined as the transmission energy per offloaded task. E^{tx}_d is the total transmission energy consumption for offloaded tasks. Similarly, $E^{receive}_j$ is defined as the energy consumed to receive the result of an offloaded task and E^{rx}_d is the total receiving energy with respect to all offloaded tasks. Finally, E^d_j is the energy consumed for performing task j locally and E^{local}_d is the total energy consumption for all tasks executed locally on device d. E^d_j is a function of the total number of CPU cycles for task j to be completed k^j_d, the processing frequency f_d of UE device d and the effective switched capacitance C_d of end device d processor. Moreover, $E^{transmit}_j$ is dependent on the UE transmit power $P^{transmit}_d$, the size of input task I_j and the uplink data rate r_{ds} between s and d. Similarly, $E^{receive}_j$ is dependent on the UE receiving power $P^{receive}_d$, the size of task result R_j and the channel downlink data rate r_{sd}. Thus, these three aforementioned equations are defined as:

$$E^d_j = k^j_d \cdot f^2_d \cdot C_d, \tag{13}$$

$$E^{transmit}_j = P^{transmit}_d \cdot \frac{I_j}{r_{ds}}, \tag{14}$$

$$r_{ds} = B_d \cdot \log_2\left(1 + \frac{P^{transmit}_d \cdot \beta^d_0}{B_d \cdot N_0 \cdot dist^2_{ds}}\right), \tag{15}$$

$$E^{receive}_j = P^{receive}_d \cdot \frac{R_j}{r_{sd}}, \tag{16}$$

$$r_{sd} = B_d \cdot \log_2\left(1 + \frac{P^{receive}_d \cdot \beta^d_0}{B_d \cdot N_0 \cdot dist^2_{ds}}\right) \tag{17}$$

where r_{sd} is a function of the UE channel bandwidth B_d, $P^{transmit}_d$, channel power gain at a reference distance 1 m, β^d_0, noise power spectrum density N_0, and $dist_{ds}$ denoting the Euclidean distance between d and s. Now, our main objective is to minimize the UE energy consumption while respecting the previously indicated constraints. The problem can thus be formulated as follows:

$$\min \sum_{d=1}^{D} E_d, \tag{18a}$$

$$\text{s.t.} \quad \mathcal{C}_1 - \mathcal{C}_7, \tag{18b}$$

It is to be noted that, in our problem, the tasks are independently assigned to the UEs, where each task has a unique arrival time and deadline. Thus, a simpler version of this problem is a generalized assignment problem (GAP) [17], which is a known NP-hard problem and therefore, our problem is also NP-hard. Resultantly, we cannot find the optimal solution in polynomial time. Therefore a meta-heuristic algorithm will be the best way to solve the problem [18].

IV. Proposed Algorithms

To address the formulated NP-hard problem, we first decompose the problem into two sub-problems including UAV trajectory design problem and task offloading problem. In this regard, **Algorithm 1** outlines the UAV trajectory mapping, which aims at providing feasible route to a UE to offload their tasks. In this context, DEA is considered to be a viable strategy to address this issue and it can reach to a sub-optimal solution [19], [20]. Thereafter, the second sub-problem is the task offloading, for which **Algorithm 2** provides static routes to UE to offload their tasks based on the EDF algorithm. Moreover, it compares the route provided from the UAV in **Algorithm 1** with the static routes previously calculated. If the UAV route is shorter than the static route, the UE will offload its task through the UAV. Otherwise, the UE will use the static route.

V. Numerical Results and Discussions

In this section, using an in house simulator, we show performance evaluation results for our proposed algorithms. As such, we set UE $= 5$ users, and we consider that BSs and the UAV are randomly distributed in a square area with the length of 1000 m. Moreover, the UAV is flying at a fixed height of 100 m and we consider 15 tasks in total, where each UE has to process 3 tasks, respectively. Furthermore, $k^j_d = 10^{-27}$ volts, f_d is set to a range of $[0, 0.8]$ GHz, C_d is set to a range of $[16, 1600]$ Hz, $P^{transmit}_d = 1$ W, whereas $P^{receive}_d = 0.2$ W, I_j is set to a range of $[10, 1000]$ KB, $B_d = 1$ dB, $\beta^d_0 = 1.42 \cdot 10^{-4}$ dB, positive constant $G_0 = 2.2846$, and $N_0 = 10^{-20}$ W/Hz. Additionally, in Fig. 2, we show the UE energy consumption of the five users under two given cases: shortest path (SP) where **Algorithm 1** is deployed without UAV assistance and shortest path with UAV

Algorithm 1: UAV task offloading using EDF

Set: Values for d, D, j, J, $dist_{ds}$, and $P_{Feasible}$, which denotes the best feasible route for set of devices D to offload their tasks J_d
for *each* d *candidate in* D **do**
  for *each* j *task in* J_d **do**
    Calculate: $dist_{ds}$ for shortest static route from d to any server s is calculated, where $dist_{ds} \in P_{static}$.
    Moreover, d includes the corresponding Euclidean distance in its own routing table. Now, server s is selected as the offload server.
    Calculate: mut_p is the dynamic route to server s, which is provided to UE by UAV once deployed using **Algorithm 1**.
    if $EuclideanDistance(mut_p) < EuclideanDistance(dist_{ds})$ *AND* $E_{min} < E_d < E_{max}$ **then**
      Set: $P_{Feasible}$ = mut_p
      Calculate: using Eqn. (6), (8) and (10) , the E2E task delay offered by UAV.
      Update: routing table of d with new route.
    end
    else
      Set: $P_{Feasible} = dist_{ds}$
      - UE rejects UAV dynamic route.
      - UAV is busy serving other UE hence mut_p provided to d is large.
    end
    Return:
    - Next hop for offloading the task.
    - Selected destination server for offloading.
  end
end

Algorithm 2: UAV trajectory mapping using DEA

Set:
- mut_p as the route to server reported by UAV deployed in a mutated location.
- T as the set of simulation time slots.
- $U(g+1)$ as the UAV new mutated location.
- $POP(X, g)$ as the population in the g^{th} generation, where X represents the UAV location. As such, we have $POP(X, g) = X(1), ..., X(g)$. Furthermore, we set $X(g)$ as the x and y coordinates of the UAV obtained in the g^{th} generation.

Generate: A random location for UAV to be $X(g)$
Set: Initial population as $POP(X, 1)$
for $t_{execute}$ *in* T **do**
  Calculate: mut_p using mutation process
  if $EuclideanDistance(mut_p) < EuclideanDistance(dist_{ds})$ **then**
    Set: $X(g+1)$ = $U(g+1)$, UAV updates location
  end
  else
    Set: $X(g+1)$ = $X(g)$, UAV stands still
  end
end

assistance (SPUA), where both **Algorithm 1** and **Algorithm 2** are used to illustrate the proposed solution. In this regard, it is clearly seen in Fig. 2 that, for all UEs, the energy consumption is the lowest in the SPUA case. This is due to the dynamic feasible routes given to the UE via the UAV. Comparably, in the SP case, high energy consumption fluctuations are seen due to unavailability of nearby BSs. Hence, UEs take the decision of offloading their tasks to farther BSs, which results in consuming more energy.

Now, the total UE energy consumption is shown in Fig. 3, where three cases are considered: **Case 1**: shortest path (SP), **Case 2**: shortest path with UAV assistance (SPUA) and **Case 3**: random assignment (RA). As the name implies, SP and SPUA consider the problem without and with UAV assistance whereas, in RA case, the UE chooses any path to offload its task without satisfying any constraints. As illustrated in Fig. 3, SPUA achieves the lowest value of energy consumption compared with the other benchmark cases or methods. Therefore, it is essential to consider UAVs for task offloading as UAVs provides better feasible paths to UEs to offload their tasks, which in turn minimizes the energy consumption. Lastly, an example of UAV trajectory is shown in Fig. 4, whereas the UAV is flying between UE in a semi-elliptical trajectory providing the UEs feasible shortest paths to the destined BSs. As such, the UAV provides such paths to the UEs, which consume a lower amount of energy.

VI. Conclusion

In this paper, we studied energy-efficient task offloading and trajectory design for UAV-based MEC systems. The goal was to minimize the UE energy consumption during task offloading process via UAV involvement, by bringing the computing capabilities near the UE via shortened paths to the available BSs. To solve the formulated NP-hard problem, we divided the main problem into two sub-problems of UAV task offloading and UAV trajectory mapping, which are solved using **Algorithm 1** and **Algorithm 2**, respectively. In this regard, **Algorithm 1** tends to provide a UE with the shortest path towards BS for offloading the assigned task, whereas **Algorithm 2** is responsible for deploying the UAV for efficient trajectory mapping in order to provide better feasible paths to the UE with respect to energy consumption. Our simulation results showed that our proposed algorithm yields better results than the benchmark methods and maps an energy-efficient UAV trajectory.

*Acknowledgement This work was supported by Mitacs/Ultra Intelligence Communications through project IT25839.

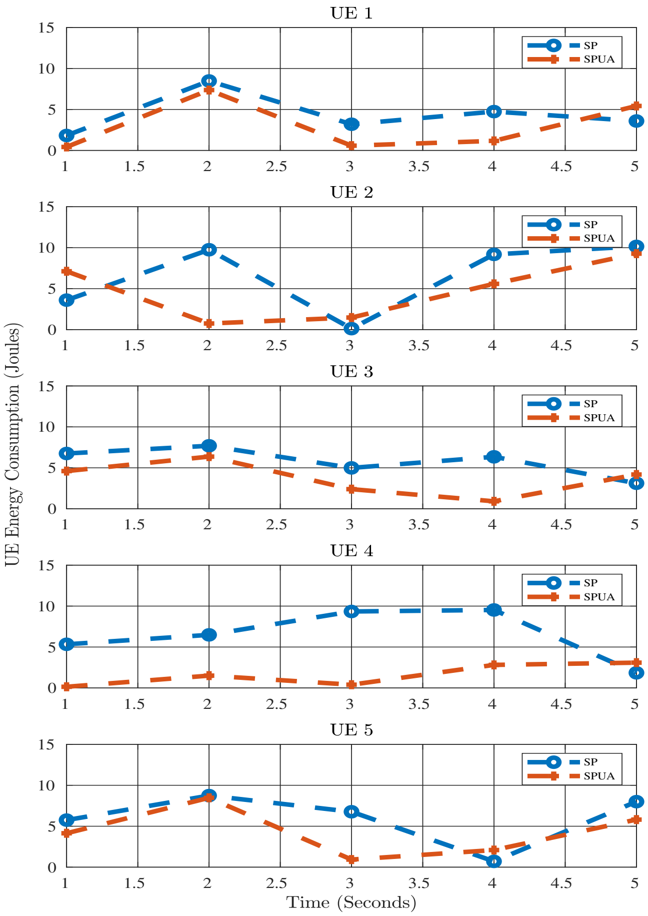

Fig. 2: UE Energy Consumption using SP and SPUA.

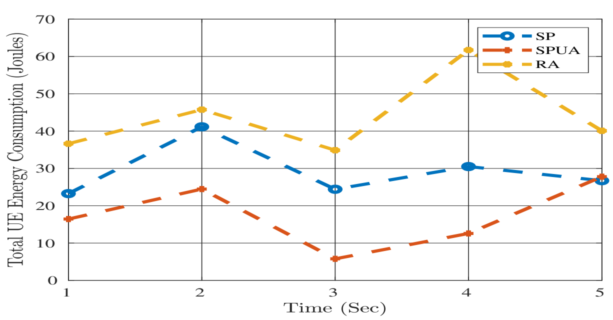

Fig. 3: UE Total Energy Consumption.

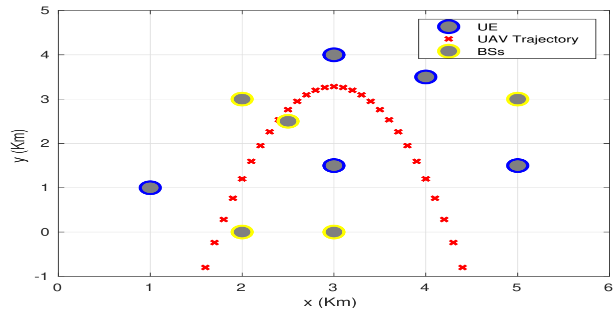

Fig. 4: UAV Trajectory for Energy-Efficient Task Offloading.

References

[1] M. Seid, G. Owusu Boateng, S. Anokye, T. Kwantwi, G. Sun, and G. Liu, "Collaborative computation offloading and resource allocation in multi-UAV assisted IoT networks: A deep reinforcement learning approach," *IEEE Internet Things J.*, vol. PP, pp. 1–1, 2021.

[2] A. Ranjha and G. Kaddoum, "URLLC-enabled by laser powered UAV relay: A quasi-optimal design of resource allocation, trajectory planning and energy harvesting," *IEEE Trans. Veh. Technol.*, vol. 71, no. 1, pp. 753–765, 2021.

[3] A. Ranjha, G. Kaddoum, M. Rahim, and K. Dev, "URLLC in UAV-enabled multicasting systems: A dual time and energy minimization problem using UAV speed, altitude and beamwidth," *Comput. Commun.*, vol. 187, pp. 125–133, 2022.

[4] K. Kim and C. S. Hong, "Optimal task-UAV-edge matching for computation offloading in UAV assisted mobile edge computing," in *20th Asia-Pacific Netw. Oper. Manag. Symp. (APNOMS)*. IEEE, 2019, pp. 1–4.

[5] S. A. Huda and S. Moh, "Survey on computation offloading in UAV-enabled mobile edge computing," *J. Netw. and Computer Appl.*, vol. 201, p. 103341, 2022.

[6] S. Shao, Y. Li, S. Guo, C. Wang, X. Chen, and X. Qiu, "Delay and energy consumption oriented UAV inspection business collaboration computing mechanism in edge computing based electric power IoT," *Chinese J. Electron.*, vol. 32, no. 1, pp. 13–25, 2023.

[7] J. Wang, K. Liu, and J. Pan, "Online UAV-mounted edge server dispatching for mobile-to-mobile edge computing," *IEEE Internet Things J.*, vol. 7, no. 2, pp. 1375–1386, 2020.

[8] S. Jeong, O. Simeone, and J. Kang, "Mobile edge computing via a UAV-mounted cloudlet: Optimization of bit allocation and path planning," *IEEE Trans. Veh. Technol.*, vol. 67, no. 3, pp. 2049–2063, 2018.

[9] F. Zhou, Y. Wu, R. Q. Hu, and Y. Qian, "Computation rate maximization in UAV-enabled wireless-powered mobile-edge computing systems," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 9, pp. 1927–1941, 2018.

[10] X. Ma, C. Yin, and X. Liu, "Machine learning based joint offloading and trajectory design in UAV based MEC system for IoT devices," in *Int. Conf. on Comput. Commun. (ICCC)*. IEEE, 2020, pp. 902–909.

[11] A. C. Nguyen, T. Pamuklu, A. Syed, W. S. Kennedy, and M. Erol-Kantarci, "Reinforcement learning-based deadline and battery-aware offloading in smart farm IoT-UAV networks," *arXiv preprint arXiv:2201.10361*, 2022.

[12] F. Sun, F. Hou, N. Cheng, M. Wang, H. Zhou, L. Gui, and X. Shen, "Cooperative task scheduling for computation offloading in vehicular cloud," *IEEE Trans. Veh. Technol.*, vol. 67, no. 11, pp. 11 049–11 061, 2018.

[13] F. S. Abkenar, S. Iranmanesh, A. Bouguettaya, R. Raad, and A. Jamalipour, "Energent: An energy-efficient uav-assisted fog-iot framework for disaster management," *Journal of Communications and Networks*, vol. 24, no. 6, pp. 698–709, 2022.

[14] R. Liu, A. Liu, Z. Qu, and N. N. Xiong, "An uav-enabled intelligent connected transportation system with 6g communications for internet of vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 2, pp. 2045–2059, 2023.

[15] W. Liu, B. Li, W. Xie, Y. Dai, and Z. Fei, "Energy efficient computation offloading in aerial edge networks with multi-agent cooperation," *IEEE Transactions on Wireless Communications*, pp. 1–1, 2023.

[16] P. A. Apostolopoulos, G. Fragkos, E. E. Tsiropoulou, and S. Papavassiliou, "Data offloading in uav-assisted multi-access edge computing systems under resource uncertainty," *IEEE Transactions on Mobile Computing*, vol. 22, no. 1, pp. 175–190, 2023.

[17] L. X. Nguyen, Y. K. Tun, T. N. Dang, Y. M. Park, Z. Han, and C. S. Hong, "Dependency tasks offloading and communication resource allocation in collaborative UAVs networks: A meta-heuristic approach," *arXiv preprint arXiv:2208.07536*, 2022.

[18] P. Chu and J. Beasley, "A genetic algorithm for the generalised assignment problem," *Comput. Oper. Research*, vol. 24, no. 1, pp. 17–23, 1997.

[19] D. Fesehaye, Y. Gao, K. Nahrstedt, and G. Wang, "Impact of cloudlets on interactive mobile cloud applications," in *Int. Enter. Distrib. Object Comput. Conf.*, 2012, pp. 123–132.

[20] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the internet of things," in *Proc. MCC Workshop Mobile cloud Comput.*, 2012, pp. 13–16.