A Game Theory Based Efficient Computation Offloading in an UAV Network

Mohamed-Ayoub Messous, Sidi-Mohammed Senouci, Hichem Sedjelmaci, *Member, IEEE*, and Soumaya Cherkaoui

***Abstract*—Recently, solutions based on mobile edge computing paradigm have been widely discussed in academia and industry. This paradigm offers solutions to address limitations, in terms of battery lifetime and processing power, of mobile and constrained devices. Despite the ever-increasing capabilities of these devices, resource requirements of applications can often transcend what is available within a single device. Offloading intensive computation tasks to a distant server can help applications reach their desired performances. In this work, we tackle the problem of offloading heavy computation tasks of unmanned aerial vehicles (UAVs) while achieving the best possible tradeoff between energy consumption, time delay, and computation cost. We focus on a scenario of a fleet of small UAVs performing an exploration mission. During their mission, these constrained devices have to carry-out highly intensive computation tasks such as pattern recognition and video preprocessing. We formulate the problem using a non-cooperative theoretical game with N players and three pure strategies. We provide a comprehensive proof for the existence of a Nash equilibrium and implement accordingly a distributed algorithm that converges to such an equilibrium. Extensive simulations are performed in order to provide thorough results and assess the performances of the approach compared to three other models. Results show that our algorithm outperforms all the three approaches. Our approach achieved in average about 19%, 58%, and 55% better results compared to local computing, offloading to the edge server, and offloading to base station, respectively.**

***Index Terms*—Mobile edge computing, computation offloading problem, non-cooperative game, pure-strategies, unmanned areal vehicles (UAVs).**

I. Introduction

UAVs (Unmanned Aerial Vehicles), aka drones, continue to attract much attention. Initially, relatively large drone platforms played a prominent role in strategic and defense programs. Recent technological advances have led to the emergence of smaller significantly cheaper UAVs which made them easier to acquire, maintain and handle, thus significantly increasing their usage in civilian applications. Indeed, UAVs proved to be useful in application like rescue missions, target detection, remote sensing, surveillance, service delivery, pollution detection and farming [1]–[9]. Drones can carry out exploration missions to replace human presence in areas which are inaccessible or hazardous. They can also deliver data to and from areas with no infrastructure [2]. In fact, thanks to the maturity of their underlying technology, along with their three-dimensional aerial mobility, drones are expected to play an enabler role in emerging networks of the future as aerial base stations to collect/deliver data from/to ground devices [3].

Even with current advances, research activities are yet to overcome some challenging issues. For instance, UAVs need to detect, classify and identify objects or situations on the spot, in order to be fully operational in surveillance applications. Besides, UAVs are brought to deal with some intensive computation tasks such as video preprocessing, pattern recognition and feature extraction. These kinds of tasks typically require executing complex algorithms and demanding calculations, which can be computation-intensive and call for dedicated and powerful processors. At the same time, limited computational power and energy supply present a major challenge for real-time data processing, networking and decision-making; all requirements of vital importance to many applications. Despite the ever-increasing capabilities of UAVs, resource requirements for applications can often transcend what is available within a single UAV. Moreover, performing intensive computation onboard an UAV may result in slow response times, can be detrimental to its battery lifetime, and ultimately can compromise mission success. In order to address issues caused by the limited resources and the intermittent connectivity in UAVs, a cloud-based solution can be adopted [10].

Several studies recommend offloading from constrained devices to remote cloud/edge servers [11]–[13]. In particular, the ability of providing unrestricted computing capabilities at the edge of the access networks for mobile devices, aka Mobile Edge Computing (MEC), is a paradigm that has received increasing attention in academic and industrial communities. Indeed, MEC was revealed as a very promising concept in order to improve network performance as well as user experience. When intensive computation tasks are offloaded to an ES significant performance enhancement can be achieved [14]–[17]. Existing research works [14]–[17] considered computation offloading to servers located either in the cloud or at the edge of the access network. The work presented in [16] suggested using a cloudlet-

Manuscript received October 5, 2018; revised January 15, 2019; accepted February 26, 2019. Date of publication February 28, 2019; date of current version May 28, 2019. The review of this paper was coordinated by Prof. J. Liu. *(Corresponding author: Mohamed-Ayoub Messous.)*

M.-A. Messous and S.-M. Senouci are with the Département De Recherche En Ingénierie Des Vehicules Pour L'environnement EA 1859, University Bourgogne Franche Comté, Nevers F58000, France (e-mail: ayoub.messous@u-bourgogne.fr; sidi-mohammed.senouci@u-bourgogne.fr).

H. Sedjelmaci is with the Orange Labs, Châtillon 92320, France (e-mail: hichem.sedjelmaci@orange.com).

S. Cherkaoui is with the INTERLAB Research Laboratory, Université de Sherbrooke, Sherbrooke, QC J4K 0A8, Canada (e-mail: soumaya.cherkaoui@usherbrooke.ca).

Digital Object Identifier 10.1109/TVT.2019.2902318

based infrastructure in order to reduce power consumption and network delay when using mobile cloud computing. Most of the previous studies only consider two choices for the offloading decision: either perform the computation locally or offload it to a distance server. Since computation services can be performed within different distant sureggate devices (servers), in this work, we intreduce a third choice for addressing the computation-offloading problem. Besides the obvious choice of locally executing a computation task, UAVs in our approach have two possible offloading choices: *(i)* they can send their tasks to a powerful nearby BS through a wireless local access network (WLAN), or *(ii)* they can send them to a more powerful server at the edge of the cellular access network (called an ES). The processing of intensive tasks will take place in one of these two remote servers and the execution results are transmitted eventually to their initiator device (UAV). We formulate this kind of decision problem using Game Theory (GT). This analytical tool has been widely used by the wireless networking community for modeling different types of problems [18]. Our objective is to optimize a global utility function, which takes into account a combination of energy consumption, delay and communication cost. We design therefore a solution that achieves the best possible tradeoff between execution time and energy overhead while taking communication cost into account. In a previous work [19], we proposed a computation offloading game for an UAV network in a mobile edge computing environment. This manuscript presents an extension of our previous paper. In contrast to the previous work, we provide an all-new design for the system model and the problem formulation. We further extend our study to cover a much generic usecase. Furthermore, a non-cooperative game with N players and 3 pure strategies is adopted to model the computation offloading problem. We also give a more comprehensive proof for the existence of equilibrium and the convergence of our distributed algorithm. We performed extensive simulation work, which provided comprehensive results to assess the performances of our approach.

This work addresses the challenges related to real-time applications where the drones are required to do computation-intensive tasks in a short amount of time. We tackle the problem of computation offloading and adopt a decentralized mechanism in which each drone makes the computation offloading decision locally. This can naturally overcome the need to implement a centralized scheme with much overhead compared to a distributed scheme. The main contributions of the present paper are as follow:

- Present a new generic approach for the computation-offloading problem using UAVs in a MEC environment.
- Conceive a non-cooperative game with N players and 3 pure strategies to model this decision problem.
- Design and implement a distributed algorithm, where decisions are made locally without a centralized entity.

The rest of the paper is organized as follows: we first summarize in Section II, research works that influenced our study. We present in Section III the system model and the formulation of our problem. Section IV gives the details of the major contributions of this paper. The simulation work and the results obtained are presented and discussed in Section V. Finally, Section VI concludes the paper and gives some future directions.

II. GT Based Computation Offloading

Mobile and distributed applications are facing a rapid growth in demand on computational and storage resources. Even though, recent technological advances have considerably improved the available resources in mobile devices, they are still not sufficient to meet the ever-growing applications demands. The most important challenges are closely related to energy management and delay minimization. On the first hand, since the energy resource is crucial for mobile devices, its optimal management is vital for network lifetime and to missions' success. On the other hand, the time required to achieve a given task is often very important. This motivated many previous studies to focus primarily on minimizing delay while optimizing energy [20]. The proposed solutions often offload computational demands to more powerful surrogate machines. The most common choice is offloading computation to a neighboring server or even to a distant cloud server through a dedicated communication interface. Such solutions have significantly succeeded in increasing computational capabilities of constrained devices. Nevertheless, overall response times may suffer considerably when many devices attempt to offload their computation tasks simultaneously. This may be primarily due to concurrent access to constrained network resources [21]. The other reason would be the size of data that needs to be offloaded, which can greatly affect transmission time as well as energy, especially for lesser complex computation tasks where it is more efficient to execute locally.

In the following, we gather a set of prominent recent works that used theoretical game methodology to tackle computation offloading problems. [22] presented a generic hybrid architecture containing a centralized cloud and a distributed MEC for an IoT environment. The authors defined a computation offloading problem and formulated their solution as collaborative game between mobile IoT devices. Authors in [23] proposed a single wireless channel to access an all-powerful cloud. They used a non-cooperative game model to implement a decentralized algorithm for offloading heavily intensive computation tasks to the cloud servers. The players of this game are all the mobile devices that have a computation task. Each player has two possible strategies: (i) local computing or (ii) offloading to the server. The authors proved the game to be a potential game and proved its convergence. Then, they extended their model in [24] to a more general use-case scenario with multiple wireless channels and showed that the game remains a potential game.

More recently, other noteworthy ideas were presented in [25], where the authors proposed a new paradigm for vehicular networks in 5G communications environment. The objective was to support data-heavy applications through a mixed-network deployment of small-cells, device-to device (D2D) and heterogeneous networks combined with cloud computing capabilities. Taking advantage of graph theory modeling, the authors demonstrate the distributed nature of their model and the relationship between cloudlets. Furthermore, they formulated a resource allocation problem via a non-cooperation matrix game and solved it though a non-linear concave optimization approach. Authors in [26] consider a dense wireless network where each individual device can offload computations via multiple access points to a mobile cloud in order to minimize their computation costs. The authors provided a game theoretical analysis of this problem while considering the set of players to be selfish. They proved the existence of a pure strategy Nash equilibrium and provided an efficient algorithm for computing an equilibrium point. The obtained simulation results showed that the equilibrium cost was close to optimal. Likewise, authors in [27] considered a multi-user mobile cloud computing system with a computing access point where each user has multiple dependent tasks to process. Since a centralized optimization solution is non-convex for the problem at hand, the authors formulated the problem as an offloading game in order to minimize the overall energy cost, computation, and the maximum delay among all users. Additionally, they showed the existence of a Nash equilibrium and proposed an algorithm to attain an equilibrium point. Finally, Yu *et al.* [28] considered a scenario where much duplicated computation tasks are processed on specific mobile users and computation results are shared through D2D multicast channels. The objective was to find an optimal network partition in order to minimize the overall energy consumption for the mobile devices. Consequently, the problem was modeled as a combinatorial optimization problem. Unlike the works mentioned previously, the authors in [28] used a different game theoretic methodology. The proposed solution was implemented using the concepts of coalitional games in order to find a maximum weighted bipartite matching. Simulation results showed a significant decrease in energy consumption while granting fairness among multiple users simultaneously.

In the same context as the related works summarized above, we tackle in our study the problem of offloading highly intensive computation tasks in mobile devices. Furthermore, we take as use-case of a fleet of small UAVs performing an exploration operation. In order to fulfill their missions, the drones are required to compute very intensive computation tasks, such as image processing, feature extraction and pattern recognition algorithms. Since, the small drones have limited computation capabilities and are powered through an onboard battery, offloading their computation tasks to a more powerful distant device would be very interesting. Nevertheless, this solution is not viable for all the possible cases, because transmission delays and the energy required to send data through the wireless medium can hinder performances. Therefore, finding the right tradeoff between energy consumption and delay is very tricky, especially in scenarios with multiple users in which the solution space can increase exponentially. To this extent, we formulate this challenging decision problem using a GT methodology as a dilemma between energy and delay. Besides, we also consider the communication cost as a third decision parameter.

As far as we know, we are the first to incorporate a combination of these three previous decision metrics, namely: energy, delay and cost, within the same utility function while also considering three different strategies rather than just two in solving the computation-offloading problem. In the following sections, we provide the details of our system model. After that, we describe a non-cooperative theoretical game where the fleet of UAVs represents the N players and each of which has three possible different strategies: (i) local computing, (ii) offload to server, and (iii) offload to BS. We prove also the existence of a Nash equilibrium where no player has the incentive to deviate from. Moreover, we design a distributed algorithm with an emerging behavior in order to reach equilibrium.

III. System Model and Problem Formulation

All mobile devices use an onboard battery with limited power. Therefore, an efficient management of the available energy supply is vital for the device's lifetime and consequently to the success of its mission. Indeed, these constrained devices are required to make the most of their available resources through their optimal usage. Energy consumption is an even more critical issue when mobile communication is required, where mobile devices need to exchange messages via a wireless channel. In this context, authors in [29] have shown that a considerable amount of energy can be saved through the usage of the right communication medium. Subsequently, they argued that since Wi-Fi technology can provide a higher data transmission rate than traditional cellular networks, Wi-Fi would yield a shorter data transmission time and therefore lower energy consumption. However, as recent LTE technology can offer a higher or comparable data rate to Wi-Fi, the energy efficient offloading problem attracts even more research interest, which is particularly true for scenarios with two possible interfaces: LTE and Wi-Fi. Many offloading schemes aiming to improve the mobile devices' energy efficiency are summarized in [30]. The most basic idea is to reduce energy consumption used in transmission while granting a bearable average transfer delay. Furthermore, the data traffic would be offloaded through a Wi-Fi access point rather than the cellular network if the difference in transmission energy exceeds a predefined threshold. In a similar context, authors in [31] considered using Wi-Fi to handle the traffic explosion problem in a vehicular network environment. They also consider reducing communication cost using roadside units to convey or download data freely rather than more expensive cellular links.

Indeed, the problem of computation offloading is very different from the data-offloading problem. In the latter, the emphasis is on data forwarding towards a distant device, as in [30] and [31], while reducing overhead. Whereas offloading computational tasks focus more on the computation and communication delay. However, both approaches try to optimize energy consumption since it is a critical resource for all mobile devices. In this context, our aim in this work is to design an optimal approach for offloading heavy computation tasks to a less constrained device. We consider both energy consumption and time delay in order to provide a comprehensive approach to solve the dilemma of jointly addressing these two criteria.

This section provides the problem formulation and highlights the system model that we have used to implement our computation offloading approach. We consider a set of drones $N = \{1, 2, \ldots, n\}$ collocated in the same area of interest (see Figure 1). In order to achieve its mission, each UAV is required to execute

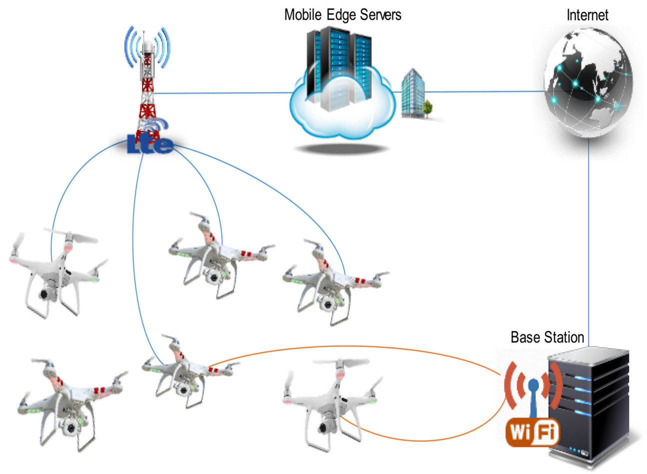

Fig. 1. System architecture.

a highly intensive and delay sensitive computation task, while preserving its available energy. Different choices are available for these constrained mobile devices. They can either: (i) perform their tasks locally, (ii) offload them via a local wireless connection to a neighboring BS, or finally (iii) through a cellular connection to an ES. The computational tasks that need to be executed are characterized by the number of CPU cycles Ci required to perform the calculation and the size of necessary data Di. Besides the input parameters necessary for the computation, the data to be sent can even include the program code if it needs to be executed remotely.

In the following subsections, we start by presenting the utility function and the different inputs required to compute its values. Then, we show the communication model used by the drones in order to communicate. Finally, we give the details related to the three possible computation models.

A. Utility Funtion

We defined our payoff function as the combination of energy consumption, delay time and communication cost. Since the vast majority of mobile devices have limited energy resources, the optimal management of this critical asset is quite beneficial for the onboard battery lifetime. Moreover, intensive computation tasks are known to necessitate a considerable amount of time to complete their execution, even with slightly powerful processors. The third entry for our performance metric is the communication cost, which would play a decisive role in choosing a suitable communication interface, because cellular networks are never freely available. Furthermore, cellular operators often charge according to the amount of data transmitted, while Wi-Fi access is perceived as a local network and are mostly free of charge. For these reasons, we implement a global payoff function as a joint equation of: (i) delay overhead, (ii) energy overhead and (iii) communication cost overhead. The resulting function for all the users is given as:

$$\boldsymbol{Utility} = \boldsymbol{\alpha} \sum_{\boldsymbol{i=1}}^{\boldsymbol{N}} \boldsymbol{T_i} + \boldsymbol{\beta} \sum_{\boldsymbol{i=1}}^{\boldsymbol{N}} \boldsymbol{E_i} + \boldsymbol{\gamma} \sum_{\boldsymbol{i=1}}^{\boldsymbol{N}} \boldsymbol{C_i} \tag{1}$$

Where: N is the number of tasks, T represents Time, E stands for Energy overhead and C is the communication Cost. Additionally, α, β and γ represent respectively the weighting parameters of delay, energy consumption and communication cost, and $\alpha + \beta + \gamma = 1$. Moreover, in order to form this global overhead metric, a specific normalization method was adopted in order to be able to add these different measures together. Furthermore, using a weighted function provide a much higher flexibility and answer a wide range of applications with specific requirements. Accordingly, depending on the envisioned application or even the current system status, different tasks can have different weighting parameters. For instance, if the device battery is running low, the value of the weight β should be increased in order to save more energy. Whereas, for a delay sensitive task, the weight α is increased in order to reduce the delay. Finally, the weight γ would to be increased or decreased according to the availability cellular communication offer.

B. Communication Models

Wi-Fi is the most recognized wireless technology that uses a set of standards for implementing a WLAN communication. It allows an interface between a wireless client and a BS or between two wireless clients. This technology has been widely used in UAV related applications [4], [32]–[35]. For instance, [32] used Wi-Fi for achieving a flight control with a real-time data such as photo and video transmission between UAVs and devices on the ground. Moreover, authors in [33] have developed an UAV-carried, on-demand Wi-Fi prototype system where the UAV carries the Wi-Fi signal to the emergency areas. While traditional Wi-Fi signal is transmitted around 100 meters, this new prototype for a UAV-carried system can extend the signal up to 25 kilometers using special directional antennas. This line of work supports the feasibility of offering Wi-Fi services through flexible UAV platforms. Another study, presented in [34], focused on enhancing Wi-Fi bandwidth for communications between UAV and ground stations. Furthermore, the results of the experimental work in [35] showed the viability of using 802.11 interfaces for UAV-based networking.

Since cellular access is widely spread, the small drones shown in Figure 1 are considered to have a cellular network access along with a second 802.11 wireless interface. This latter is used to access the BS while the connection to the ES is achieved through a cellular network (3G/LTE). We denote $d_i \in \{0, 1, 2\}$ as the computation offloading decision for node i. Explicitly, we have $d_i = 0$ if n chooses to compute its task locally. Otherwise, we would have $d_i = 1$ or $d_i = 2$ if it chooses to offload the computation, respectively, to the ES via the cellular network or to the BS via *Wi-Fi* connectivity. It worth mentioning that if too many devices choose to offload their computation tasks simultaneously, via the same medium, some severe interference may incur. It leads subsequently to low data rates, which would negatively affect the overall network performances. In a controlled environment and knowing the decisions of all the mobile devices $(d_1, d_2, \ldots, d_N)$, it is eventually possible to compute the conceivable data rates for each node. Moreover, communication delay contains the network propagation delay and the data transmission delay. In our case, P, the network propagation delay, is determined by transmission distance, while the data transmission delay is jointly determined by $\mathrm{D_i}$, the amount of data being trnsfered, and $\mathrm{R_i}$, the link badwidth.

C. Computation Model

For the computation aspects of the proposed approach, we consider that each device has one or more computation intensive tasks. Each task T_i is defined through $(C_i,\, D_i)$ which are respectively the number of computation cycles required to achieve a result and the size of data that needs to be forwarded. Data can include the input parameters necessary for the computation and the program code to be executed. The devices have to decide whether to execute their computation tasks either locally or offload them to a remote station. Three possible choices are available and for each of which a different value for the utility function is assumed. The details related to how to compute these values are given in the following paragraphs.

1) Local Computing: Since computation tasks in the "local computing" are executed locally, no actual data ought to be sent via wireless interfaces. Therefore, the utility function would only be impacted by the computation power available in the device, i.e., the CPU frequency which is the number of computation cycles per a time unit. Subsequently, the execution time for a task $T_i = (C_i,\, D_i)$ if the local CPU frequency is F_{CPU}^{Local} is given as:

$$\boldsymbol{T_{Local} = C_i / F_{CPU}^{Local}} \tag{2}$$

And as for the expected energy consumption, we would have:

$$\boldsymbol{E_{Local} = C_i * e_{CPU}^{Local}} \tag{3}$$

Where e_{CPU}^{Local} is the coefficient value representing the energy consumed per CPU cycle.

2) Offloading to the ES: The first possible offloading approach is to send the computation task via a compatible cellular access network to the ES. This latter will compute the received task instead of the mobile node. Compared to the previous option, the delay or time required to obtain results for the task being executed, in addition to the computation time, will incur an extra overhead. This is due to the additional time necessary to transmit data up to the ES. Therefore, the equation for the time function would be written as:

$$\boldsymbol{T_{ES} = C_i / F_{CPU}^{ES} + D_i / R_{Cellular}} \tag{4}$$

Where F_{CPU}^{ES} represents the frequency of the server CPU, which in practice is very big compared to the running frequency for the mobile devices' CPUs. $\mathrm{R_{Cellular}}$ is the effective data rate achieved through the cellular network as given in section III.A.

When compared to mobile devices, energy resource is abundantly available for the server. So, for the energy cost required to achieve a computation task we only consider the energy required for its transmission to the ES. Thus, the energy function is given as:

$$\boldsymbol{E_{ES} = D_i * e_{cellular}} \tag{5}$$

Where $\mathrm{e_{cellular}}$ denotes the consumption coefficient required to send one unit of data through the cellular network to the ES.

$$\boldsymbol{C_{ES} = D_i * c_{cellular}} \tag{6}$$

Where $\boldsymbol{c}_{cellular}$ represents the communication cost required to send one unit of data through the cellular network to the ES.

3) Offloading to the BS: The third possible choice and the second offloading approach considered in this work is to offload the computation task through a wireless access point to a nearby BS. This latter would compute the received task on behalf of the mobile node. In this case, the time delay and energy cost are given respectively as:

$$\boldsymbol{T_{BS} = C_i / F_{CPU}^{BS} + D_i / R_{wi-fi}} \tag{7}$$

and:

$$\boldsymbol{E_{BS} = C_i * e_{CPU}^{BS} + D_i * e_{wi-fi}} \tag{8}$$

Where F_{CPU}^{BS} denotes the CPU's frequency of the BS, R_{wi-fi} is the effective data rate achieved through the WLAN and e_{CPU}^{BS} measures the energy required to execute one CPU cycle. Finally, e_{wi-fi} represents the coefficient measuring the energy needed to send one data unit through the available access point network to the BS.

As many previous studies [23], [24], [26], we neglect the delay overhead required to send back the computation result to its respective initiator. This is due to the fact that the size of data resulting from an intensive computation task is considered very small and eventually insignificant compared to the size of the input data. This assumption holds for many scenarios such as video processing, feature extraction and pattern recognition algorithms, where the program codes and input parameters size are much bigger than the input data.

In the following sections and using the system model presented above as a guideline, we will develop a decentralized algorithm based on a GT approach for offloading highly intensive computation tasks to more powerful and less constrained network nodes.

IV. GT Based Distributed Computation Offloading Strategy

We tackle throughout our study the issue of implementing an efficient decentralized algorithm for the offloading of heavy computation tasks either to an ES or to a neighboring BS. From the computation and communication models presented in the previous section, it is clear that the decisions made by the drones are highly coupled. This means that each locally made decision will have a direct impact on the other devices evolving in the same system. Furthermore, if a significant number of drones choose simultaneously the same offloading strategy through the same access network, it would have a direct impact on the network performances, which would subsequently affect transmission data rate. Low network throughput would lead to higher transmission delay. Moreover, when the data rate is low, much more energy would be consumed to offload data. In order to get around this issue, it would be more profitable to use another offloading strategy or even choose a local computation. In the aim of achieving the best possible computation offloading decision, we design and implement a decentralized algorithm based on a non-cooperative theoretical game model.

A. Computation Offloading Game

GT is a powerful tool to analyze the interactions between multiple independent entities that are required to work together in order to achieve their own goals. Using practical examples, authors in [18] have shown how GT can be used as an enabling tool in resolving wireless networking problems. They also summarized the basic notions of non-cooperative games, where the players are the devices in the network. The first reason for choosing GT as an enabling framework for our approach is the decentralized nature of the decision-making process achieved by the network nodes. Since each entity may have different requirements and eventually does not pursue the same interest, a decentralized scheme is required where each player chooses the best possible strategy to achieve its own goals. Thus, decentralized schemes are formulated with low complexity by leveraging the intelligence of each individual device. The other reason is the complexity and viability of implementing a centralized approach. The optimization problem that we are facing is fundamentally difficult. The authors in [24] proved that the cardinality for this resource allocation problem is similar to the bin-packing problem, which is known to be NP-hard [36].

In GT, players ought to self-organize into a mutually satisfactory solution such that no player has the incentive to deviate unilaterally. Therefore, it would ease the burden of implementing a more complex centralized system. This mutually satisfactory solution where no player has the incentive to unilaterally change its strategy is called a Nash Equilibrium [37]. In order to obtain meaningful insights from our analysis study, we make the common assumption that the number of drones does not change during a mission [38], [39]. Similar to many previous studies dealing with mobile cloud computing [40] and mobile edge computing [23], [24], we consider a quasi-static scenario where the nodes positions remain unchanged, during a period of time, while they may move throughout different periods.

In this study, we use a strategic form non-cooperative game denoted as $\mu(N,S,U)$, which consists of: *(i)* a finite set of players representing the set $N = \{1, \ldots\ N\}$, where N is the number of drones in the fleet, *(ii)* Strategy space S, representing the set of actions that each player can take. $S = \{s_1, s_2,\ s_3\}$, *(iii)* Utility function Ui for each player i. The strategy selected by the UAVj could be either local computing, offload to the ES or offload to the BS. Therefore, S_i can be defined as $S_i = \{s_j; \forall j\ \in\ (0 \triangleq\ Local\, computing,\ 1 \triangleq Offload\, to\, ES,\ 2 \triangleq Offload\, to\, BS)\}$. The strategies of all the other players excluding UAV_i are denoted S_{-j}. The utility function of UAV_i represents the overhead generated through one of the previous strategies while taking into account the strategies of all the other players. This is called a strategy profile and is denoted $(S_j,\ S_{-j})$. Details about the possible values of Ui are shown in Eq. (9).

$$\boldsymbol{U_i\,(s_j, S_{-j})} = \begin{cases} \boldsymbol{U_{Local} = \alpha E_{Local} + \beta T_{Local} + \gamma C_{Local}} & \textbf{if } \boldsymbol{s_i = 0} \\ \boldsymbol{U_{ES} = \alpha E_{ES} + \beta T_{ES} + \gamma C_{ES}} & \textbf{if } \boldsymbol{s_i = 1} \\ \boldsymbol{U_{BS} = \alpha E_{BS} + \beta T_{BS} + \gamma C_{BS}} & \textbf{if } \boldsymbol{s_i = 2} \end{cases} \tag{9}$$

In this type of strategic form games, the underlying assumption is that preferences of players are captured through the utility functions, i.e., the strategy profile $(s\prime_j,\ S\prime_{-j})$ is more profitable than the strategy profile $(s'_j,\ S'_{-j})$ for the player i if and only if $U_i\prime(s_j\prime,\ S_{-j}\prime) > U'_i(s'_j,\ S'_{-j})$. Moreover, players in our game are assumed to be non-cooperative, such that each player acts independently to improve its own utility function. They are also considered to be rational in the sense that they utilize strategies with better utility. These three assumptions lead eventually to an equilibrium state for all the players called the Nash Equilibrium (*NE*), where no player has the incentive to deviate unilaterally [41], [42].

B. Nash Equilibrium

In order to achieve a stable convergence state in a non-cooperative game, all the players need to reach a common consensus status, namely a Nash Equilibrium. This optimal state represents a stable point where no player has the incentive to deviate from. This means that no player can further improve his utility function by unilaterally changing his strategy. Furthermore, a *NE* is a strategy profile from which no player can unilaterally deviate and improve its payoff [41], [42]. *NE* represents a stable outcome for a strategic form game. When equilibrium is reached, rational players cannot deviate from this strategy profile. This makes *NE* one of the most frequently used solution concepts for games. A formal definition is provided below.

Definition 1 (Nash Equilibrium):

$$A\ strategy\ profile\ S^* : (s_1^*, \ldots s_N^*)\, is\ \ a\ Nash\ equilibrium$$

$$\Leftrightarrow U_i^*\left(s_j^*, S_{-j}^*\right) \leq U_i\left(s_j, S_{-j}^*\right)\ \forall\ i\ \in N,\ \forall\ s_j \in S$$

Where N is the number of players participating in the game, S is the set of the strategies of each player and U_i is the value of the utility function defined in eq. (9).

In order to proof the existence of a *NE* and eventually the convergence of our game, we resort to the concept of potential games, presented for the first time in [43]. Since a potential game has at least one *NE* solution [23], [24], we can prove whether our offloading game may achieve a *NE*. Potential games are defined as non-zero-sum games in which the determination of a *NE* can be equivalently posed as the optimization by all the players of a single function, called a potential function. This latter is used for analyzing equilibrium properties of games because all players' objectives are aligned with a global objective. Furthermore, potential games are special class of theoretical games in which all players' preferences are coupled into the same function. This feature is profitable since it makes potential games easier to analyze and it also ensures the convergence of the game to an equilibrium through simple dynamics. A formal definition for a potential game is given in the following.

Definition 2 (Potential Game): A game μ(N,S,U) is a potential game if there exist a potential function P:S→R such that, for all i∈N, all $s_{-j} \epsilon S_{-j}$ *and* $s_j,\ s\prime_j\ \epsilon S_j,\ U_i\left(s_j,\ S_{-j}\right) - U_i\left(s\prime_j,\ S_{-j}\right)) = P\left(s_j,\ S_{-j}\right) - P\left(s\prime_j,\ S_{-j}\right)$

Now, in order to proof the convergence of our approach, we need to proof the existence of a *NE*. Since, it has been already proven that every potential game has a *NE* [23], [24], [44], we only need to prove that our game *μ(N,A,G)* is a potential game.

Proof: A game G is a potential game if its utility function U can be expressed as a Potential function $P\left(s_j,\ S_{-j}\right)$. *This latter is defined as.*

$$P\left(s_j,\ S_{-j}\right) - P\left(s'_j,\ S'_{-j}\right) = U_i\left(s_j,\ S_{-j}\right) - U_i\left(s'_j,\ S'_{-j}\right)$$

$$P\left(s_j,\ S_{-j}\right) = arg\ min_{s_j \in S_i} U_i\ \left(s_j, S_{-j}\right) = \psi_i\left(S_{-j}\right);$$

$$P\left(s'_j,\ S'_{-j}\right) = \psi_i\left(S'_{-j}\right);$$

Where $\psi_i(S_{-j})$ is the best possible payoff for a player i given the strategy profile S_{-j}. It is also defined as a best-response potential game [44], which is equal to Eq. (10).

$$\boldsymbol{\psi_i\left(S_{-j}\right)} = \begin{cases} \boldsymbol{arg\ min_{s_j \in S_i} U_{Local}}, & \boldsymbol{if\ s_i = 0} \\ \boldsymbol{arg\ min_{s_j \in S_i} U_{ES}}, & \boldsymbol{if\ s_i = 1} \\ \boldsymbol{arg\ min_{s_j \in S_i} U_{BS}}, & \boldsymbol{if\ s_i = 2} \end{cases} \quad (10)$$

μ(N,S,U) is a potential game since Eq. (10) satisfies the definition of a potential function and it provides an optimal solution that ensures the best tradeoff between a low overhead and achieving the requested task. Therefore, the *NE* solution is unique and it is equal to $\psi_i(S_{-j})$. In the real experiment, the players choose the strategy that corresponds to the minimum from the three possible values of *U:* $arg\ min_{s_j \in S_i} U_{Local}$, $arg\ min_{s_j \in S_i} U_{ES}$ *and* $arg\ min_{s_j \in S_i} U_{BS}$.

C. Decentrelized Offloading Algorithm

The analysis study provided above shows the stable profile for the player's decisions when the equilibrium is reached. Nevertheless, a decentralized algorithm is required to implement our distributed computation-offloading scheme and eventually enable the UAVs to attain a mutually satisfactory goal. The main idea behind our algorithm is to use the convergence property reached thanks to the *NE* theorem presented in the previous section. Since, a finite number of iterations is needed to achieve this plateau status. The decision-making process is executed simultaneously all over the network devices before launching the computation tasks. Similar algorithms have been proposed for cloudlet-based [16] and mobile cloud computing [23], [24]. To implement the concurrently selfish behavior of the different players, we proposed in our case a simple message exchange protocol. In the latter, each drone initiates a request message to update its status if a better strategy is attainable. Nonetheless, at each iteration, one single update request is approved via an acknowledgment message, so that only one decision is made at a time. The flowchart in Figure 2 summarizes the main steps of the proposed algorithm.

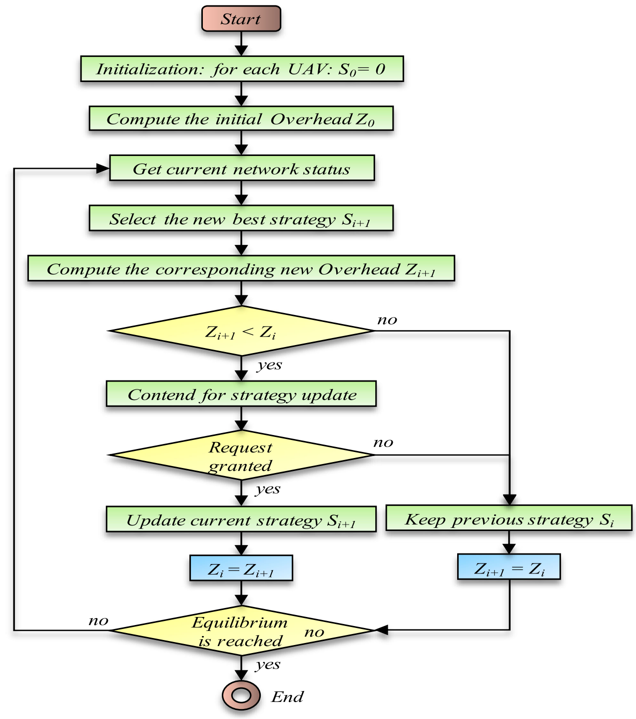

Fig. 2. Computation offloading algorithm.

For higher number of nodes, the concurrent nature of our scheme and the number of iterations required to reach the equilibrium might eventually rise a scaling problem, especially in very dense networks. This issue can be handled by adopting a hierarchical multi-tier scheme in order to resolve the contention [45]. However, it should be noted that, based on the hypothesis initially presented in the system model section, the proposed scheme as it is currently presented does not suffer from this scaling issue for the following reasons: (i) Only a limited number of UAVs can be used at the same time in the same exploration mission. (ii) In order to offload tasks to the BS, we only consider one wireless access point serving a predetermined number of users in the same region (as shown in figure 1).

V. Experimental Results and Discussions

We evaluate, in this section, the performances of the approach based on a theoretic game compared to three other strategies: (i) Local Computing, (ii) Offloading to ES, and (iii) Offloading to BS. In the first model, all the computation tasks are executed locally. However, they are offloaded to ES via a cellular access network in the second approach and to the BS via a WLAN in the third approach. We evaluated the system-wide overhead, which is defined in the previous sections as a combination of delay overhead, energy overhead and communication cost. Different scenarios are considered in our simulation while varying each time the possible entries. Specifically, we evaluate the impact of the size of the UAV network on the performances by changing the number of drones. Furthermore, since the global overhead is directly affected by the size of data that need to be transmitted and the CPU cycles required of computing a task, these two

TABLE I
SIMULATION SCENARIOS

Scenarios		Inputs
UAVs #		[5, ... 50]
Tasks	Ci	[5, … 100] (x105)
	Di	[10, … 200] (x103)
(α , β, γ)		(2/5, 2/5, 1/5)

TABLE II
SIMULATION PARAMETERS

Parameters	Values
$(F^{Local}_{CPU}, F^{ES}_{CPU}, F^{BS}_{CPU})$	(1, 30, 5) GHz
$\left(e^{Local}_{CPU}; e^{ES}_{CPU}; e^{BS}_{CPU}\right)$	(2, 0, 1) units
$(R_{Cellular}; R_{Wi\text{-}Fi})$	(1; 10) Mbps
$(e_{Cellular}, e_{Wi\text{-}Fi})$	(1200; 1000) units
$(Cost_{Cellular}, Cost_{Wi\text{-}Fi})$	(1; 0) unit

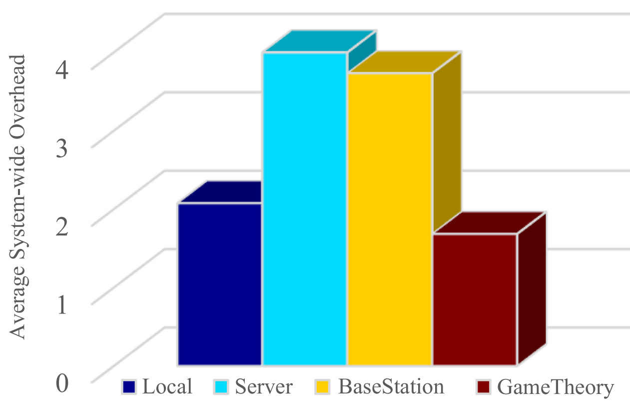

Fig. 3. Average system-wide overhead.

parameters need to be taken into consideration. Therefore, we study also the impact that data sizes and computation cycles may have on the overall overhead. The set of simulation scenarios and the main simulation parameters are summarized respectively in Table I and Table II.

We consider F^{ES}_{CPU} the CPU capability of the ES to be six times more powerful than BS frequency F^{BS}_{CPU}, which is five time more powerful than the local processing frequency F^{Local}_{CPU} available within the drone. As for energy coefficients, we selected realistic parameters similar to those used in the literature [46]–[49]. We consider that sending one data unit to the BS through the WLAN interface e_{wi-fi} consumes more than computing one CPU cycle locally e^{Local}_{CPU} [48]. This latter is twice the energy consumed if the calculation is executed in the BS e^{BS}_{CPU}. As for the energy coefficient of the server, we consider it unlimited, thus $e^{ES}_{CPU} = 0$. Furthermore, we consider that the use of cellular network to offload data consume 20% more energy than Wi-Fi, therefore, $e_{Cellular}$ and e_{wi-fi} needs 1200 units and 1000 units respectively to send a single packet of data [46], [49]. Finally, to introduce the communication cost in our simulation, we consider that Wi-Fi is free ($Cost_{wi-fi} = 0$) whereas cellular network is not ($Cost_{cellular} = 1$ unit).

The diagram shown in Figure 3 represents the average system wide overhead. It reveals that our approach outperforms the three models in terms of global overhead. This is due to the fact that our model always chooses the most efficient strategy while taking time overhead, energy consumption and communication cost into consideration.

Moreover, in order to investigate the impact of the network size on the model performances, we evaluate scenarios with different number of UAV. The results shown in Figure 4(a) represent the system-wide overhead achieved, when we vary the network size, through our model compared to the three different strategies. We can notice the continuous growth in the values of global overhead at the same time with the size of the network. This is quite normal, since with the increase of the number of drones, the number of computation tasks increases accordingly, therefore, much more resources are needed which would translate in greater values for the system-wide overhead. Nevertheless, within the same graph, the values achieved through the theoretical game approach were always better than the three other models. We also considered, in our evaluation study, the impact that different computation cycles have on our approach. Thus, we performed new simulations while fixing the number of CPU cycles Ci for each scenario and changing the size of data and the number of drones. We than calculated the average values for the system-wide overhead that correspond to each computation cycles. Results are shown in Figure 4(b). We can see that the average system overhead achieved through our theoretical game approach outperforms the three other models in all the considered scenarios. The achieved results were even better for highly intensive computation tasks. Indeed, the average system overhead for the theoretical game model increases much slower, compared to other models, because as the number of processing cycles increases, more UAVs choose to offload their tasks to mitigate the computation delays of local computing. Additionally, Figure 4(b) also shows that the local computing is most suitable for less intensive computation tasks, namely for values that are less than 5x106 CPU cycles. Inversely, offloading to server is more appropriate for highly intensive tasks with more than 10x106 CPU cycles. Between these two intervals, local computing and both offloading strategies achieved closer performances.

Furthermore, to evaluate the impact of data sizes that computation tasks need to send, we finally run different simulations with the same data sizes while changing each time the computation cycles and the number of UAVs. Figure 4(c) shows that the system-wide overhead increases as the data size increases in the two offloading approaches, due to the fact that big data induce high transmission overhead. Nevertheless, system overhead in the theoretical game approach increases slowly when data size increases. This is because more UAVs choose to avoid the heavy cost of offloading via wireless interfaces and compute their tasks locally. In this last case, the size of data has a direct impact on communication cost, communication delay and even the amount of energy required for packet's transmission using the cellular link. Moreover, the local computing approach delivers much better results compared to the offloading strategies for computation task that require transmitting more than 50×103 packets. Whereas, offloading approaches always outperform the local approach when the size of data is less than 25×103 packets.

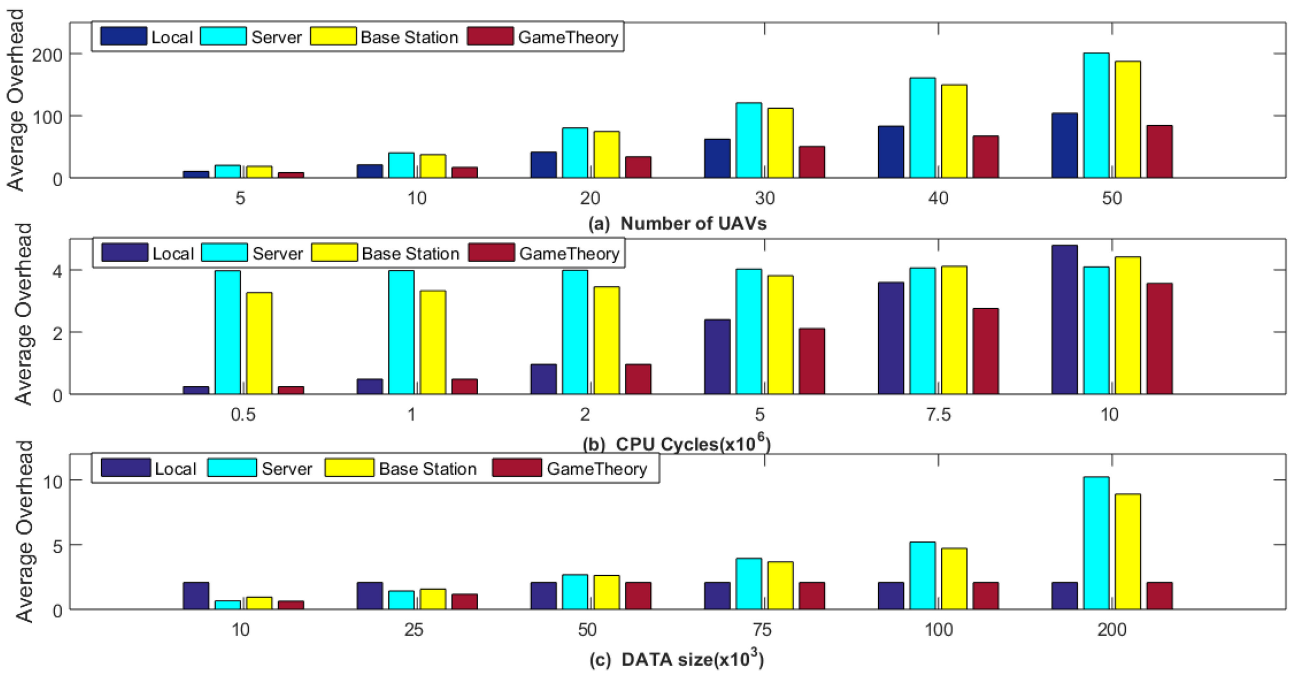

Fig. 4. Impact of different evaluation parameters on the average overhead.

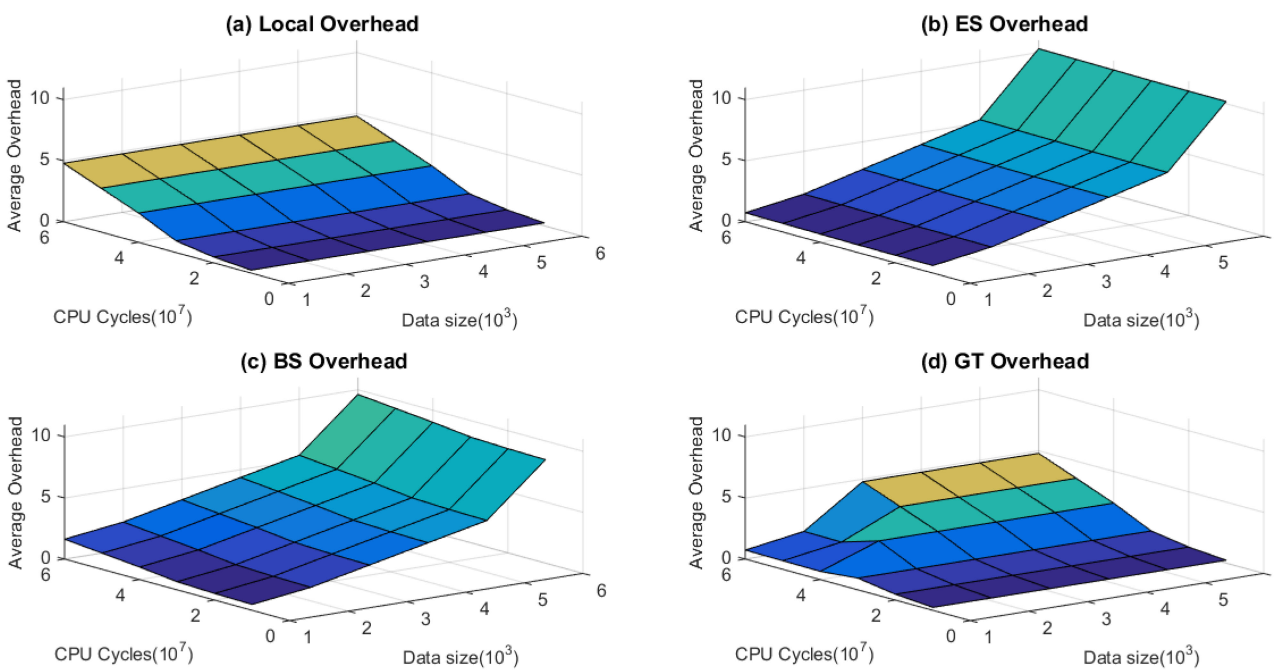

Fig. 5. System wide average overhead.

The results shown in the previous figures only provide a unique angle each time. For a more comprehensive analysis of our simulation results, we consider the impact that changing data sizes and computation cycles at the same time may have. This duality is thoroughly examined in Figure 5 and 6, where we evaluate respectively the average values of overhead for the different approaches than delay and energy for GT model. Figure 5 confirms that the GT based approach (Figure 5(d)) clearly outperforms the three other models in terms of average overhead. Furthermore, we notice that values of average overhead in the local computing approach (Figure 5(a)) are more correlated with the computation intensity of tasks. Consequently, this means that the data size does not have any impact of the overall overhead since tasks do not require to be transferred to a distant device. Whereas, both of the offloading strategies in Figure 5(b) and Figure 5(c) are more affected by data sizes

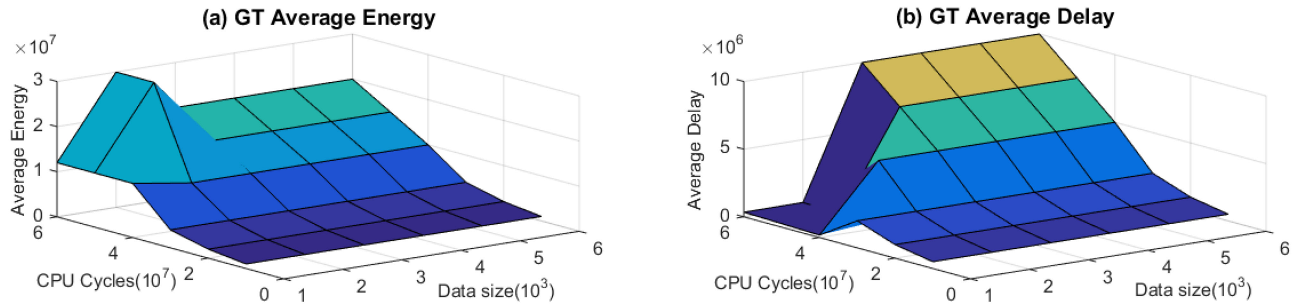

Fig. 6. Evaluation of average delay and average energy achieved by GT based apporach.

rather than the required computation cycles. Nevertheless, since ES has more computation power than BS, CPU cycles have less impact on the overhead when offloading to ES compared to offloading to BS. Whereas, inversely data size influences more transmitting tasks to ES rather than sending data to BS. This is because cellular network consumes more energy compared to Wi-Fi while providing less data rate.

Since the overhead function comprises disjoint parameters, a global overview is still missing. In order to fully examine the performances of the proposed model, we provide even more comprehensive analysis in Figure 6. Specifically, we assess the impact of the GT based approach on the communication and computation time in Figure 6(a) and its impact on energy consumption in Figure 6(b). It can be seen that a stronger correlation between computation complexity, expressed in CPU cycles, with average energy consumption and delay compared to data sizes. However, the size of data still has an important impact of the offloading decision. As a summery, we can say that the GT based approach made its offloading decisions based on the most efficient choice based on global overhead expressed in communication cost, time delay and energy consumption.

VI. Conclusion

Thanks to the recent technological advances, UAVs are currently emerging as versatile nascent paradigm that can be used in exploration and surveillance missions. However, the corresponding span of applications requires very often complex computing in a limited amount of time. Nonetheless, on the one hand, due to the limited computation and energy resources available within UAVs, time delay and energy consumption for these constrained devices are still a major challenge. On the other hand, services and functionalities offered through the concept of mobile edge computing (MEC) provide feasible alternatives to mitigate the issues facing these constrained and mobile devices. In this paper, we consider the problem of offloading highly intensive computation tasks in a fleet of small UAVs to decrease the execution delay while optimizing the energy overhead. We formulate the problem using a non-cooperative theoretical game with N players and three pure strategies, which are: (i) local computing, (ii) offload to an ES, or (iii) offload to a powerful BS. Additionally, we define an all-new utility function that combines energy overhead, computation and communication delays while taking the communication cost into account. We also provide a comprehensive proof for the existence of a *NE* and implement accordingly a distributed algorithm that converges to such an equilibrium. To gauge the effectiveness of our proposal, extensive experimental work was achieved. Simulation results show that our model outperforms other approaches, provides better performances and significantly reduces the average system-wide overhead.

As future direction to our work, we intend to implement and assess the performances of our computation-offloading approach through a cooperative game. We plan also to further evaluate the impact that the weighting parameters used in our utility function may have on the overall overhead. In the same context, considering a dynamic selection of the weighting parameters, depending on the requirements of each computation task, would make our scheme even more generic and provide additional setting for the final user.

References

[1] G. Pajares, "Overview and current status of remote sensing applications based on unmanned aerial vehicles (UAVs)," *Photogrammetric Eng. Remote Sens.*, vol. 81, no. 4, pp 281–329, Apr. 2015.

[2] L. Ma, M. Li, L. Tong, Y. Wang, and L. Cheng, "Using unmanned aerial vehicle for remote sensing application," in *Proc. 21st Int. Conf. Geoinformatics*, Kaifeng, China, 2013, pp. 1–5.

[3] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Mobile unmanned aerial vehicles (UAVs) for energy-efficient Internet of Things communications," *IEEE Trans. Wireless Commun.*, vol. 16, no. 11, pp. 7574–7589, Nov. 2017.

[4] N. Hossein Motlagh, T. Taleb, and O. Arouk, "Low-altitude unmanned aerial vehicles-based Internet of Things services: Comprehensive survey and future perspectives," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 899–922, Dec. 2016.

[5] M. A. Messous, S. M. Senouci, and H. Sedjelmaci, "Network connectivity and area coverage for UAV fleet mobility model with energy constraint," in *Proc. IEEE Wireless Commun. Netw. Conf.*, Doha, Qatar, 2016, pp. 1–6.

[6] M. A. Messous, H. Sedjelmaci, and S.-M. Senouci, "Implementing an emerging mobility model for a fleet of UAVs based on a fuzzy logic inference system," *Elsevier Pervasive Mobile Comput.*, vol. 42, pp. 393–410, 2017.

[7] C. A. Kerrache, A. Lakas, N. Lagraa, and E. Barka, "UAV-assisted technique for the detection of malicious and selfish nodes in VANETs," *Elsevier Veh. Commun.*, vol. 1, pp. 1–11, 2017.

[8] O. S. Oubbati, A. Lakas, F. Zhou, M. Güneş, and M. B. Yagoubi, "A survey on position-based routing protocols for flying ad hoc networks (FANETs)," *Elsevier Veh. Commun.*, vol. 10, pp. 29–56, 2017.

[9] J. Liu, Y. Shi, Z. Fadlullah, and N. Kato, "Space-air-ground integrated network: A survey," *IEEE Commun. Surv. Tut.*, vol. 20, no. 4, pp. 2714–2741, Forth Quarter 2018, doi: 10.1109/COMST.2018.2841996.

[10] C. Luo, J. Nightingale, E. Asemota, and C. Grecos, "A UAV-cloud system for disaster sensing applications," in *Proc. IEEE 81st Veh. Technol. Conf.*, Glasgow, U.K., 2015, pp. 1–5.

[11] A. Mtibaa, K. A. Harras, K. Habak, M. Ammar, and E. W. Zegura, "Towards mobile opportunistic computing," in *Proc. IEEE 8th Int. Conf. Cloud Comput.*, New York City, NY, USA, 2015, pp. 1111–1114.

[12] E. Cuervo *et al.*, "MAUI: Making smartphones last longer with code offload," in *Proc. 8th Int. Conf. Mobile Syst. Appl. Services*, Jun. 2010, pp. 49–62.

[13] H. Guo, J. Liu, and J. Zhang, "Efficient computation offloading for multi-access edge computing in 5G HetNets," in *Proc. IEEE Int. Conf. Commun.*, Kansas City, MO, USA, 2018, pp. 1–6.

[14] A. Ahmed and E. Ahmed, "A survey on mobile edge computing," in *Proc. IEEE Int. Conf. Intell. Syst. Control*, 2016, pp. 1–8.

[15] M. T. Beck, M. Werner, S. Feld, and T. Schimper, "Mobile edge computing: A taxonomy," in *Proc. 6th Int. Conf. Adv. Future Internet*, 2014, pp. 48–54.

[16] Y. Jararweh, L. Tawalbeh, F. Ababneh, and F. Dosari, "Resource efficient mobile computing using cloudlet infrastructure," in *Proc. IEEE 9th Int. Conf. Mobile Ad-Hoc Sensor Netw.*, Dalian, 2013, pp. 373–377.

[17] M. A. Messous, A. Arfaoui, A. Alioua, and S. M. Senouci, "A sequential game approach for computation-offloading in an UAV network," in *Proc. IEEE Global Commun. Conf.*, Singapore, Dec. 2017, pp. 1–7.

[18] M. Felegyhazi and J-P. Hubaux, "Game theory in wireless networks: A tutorial," EPFL, Lausanne, Switzerland, LCA-Rep.2006-002, 2006.

[19] M. A. Messous, S. M. Senouci, and H. Sedjelmaci, "Computation offloading game for an UAV network in mobile edge computing," in *Proc. IEEE Int. Conf. Commun.*, Paris, France, 2017, pp. 1–6.

[20] M. R. Rahimi, J. Ren, C. H. Liu, A. V. Vasilakos, and N. Venkata- subramanian, "Mobile cloud computing: A survey, state-of-art and future directions," in *Proc. ACM/Springer Mobile Netw. Appl.*, 2013, pp. 1–11.

[21] M. A. Hassan, M. Xiao, Q. Wei, and S. Chen, "Help your mobile applications with fog computing," in *Proc. 12th Annu. IEEE Int. Conf. Sens., Commun.,Netw.Workshops*, Seattle, WA, USA, 2015, pp. 1–6.

[22] H. Guo, J. Liu, and H. Qin, "Collaborative mobile-edge computation offloading for IoT over fiber-wireless networks," *IEEE Netw. Mag.*, vol. 32, no. 1, pp. 66–71, Jan./Feb. 2018.

[23] X. Chen, "Decentralized computation offloading game for mobile cloud computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 4, pp. 974–983, Apr. 2015.

[24] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation of floading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, Oct. 2015.

[25] R. Yu, J. Ding, X. Huang, M.-T. Zhou, S. Gjessing, and Y. Zhang, "Optimal resource sharing in 5G-enabled vehicular networks: A matrix game approach," *IEEE Trans. Veh. Technol.*, vol. 65, no. 10, pp. 7844–7856, Oct. 2016.

[26] S. Josilo and G. Dán, "Selfish decentralized computation offloading for mobile cloud computing in dense wireless networks," *IEEE Trans. Mob. Comput.*, vol. 18, no. 1, pp. 207–220, 2019.

[27] M. H. Chen, M. Dong, and B. Liang, "Multi-user mobile cloud offloading game with computing access point," in *Proc. 5th IEEE Cloudnet*, Pisa, 2016, pp. 64–69.

[28] S. Yu, R. Langar, and X. Wang, "A D2D-multicast based computation offloading framework for interactive applications," in *Proc. IEEE Global Commun.Conf.*, Washington, DC, USA, 2016, pp. 1–6.

[29] K. Lee, J. Lee, Y. Yi, I. Rhee, and S. Chong, "Mobile data offloading: How much can WiFi deliver?," *IEEE/ACM Trans. Netw.*, vol. 21, no. 2, pp. 536–550, Apr. 2013.

[30] N. Cheng, N. Lu, N. Zhang, X. S. Shen, and J. W. Mark, "Vehicular WiFi of floading: Challenges and solutions," *Elsevier Veh. Commun.*, vol. 1, no. 1, pp. 13–21, Jan. 2014.

[31] N. Wang and J. Wu, "Opportunistic WiFi offloading in a vehicular environment: Waiting or downloading now?," in *Proc. 35th Annu. IEEE Int. Conf. Comput. Commun.*, San Francisco, CA, USA, 2016, pp. 1–9.

[32] C. Zhu, X. Wang, and Z. Liu, "Commercial UAV positioning use case," IEEE Document: IEEE 802.11- 15/0907r1. Huawei Technologies, Jul. 2015.

[33] S. Fu and Y. Wan, "Spotlight: UAVs for disaster area communication," HDIAC, Jul. 2015. [Online]. Available: https://www.hdiac.org/

[34] J. Chen, K.-F. Tong, and J. Wang, "A triple band arc-shaped slot patch antenna for UAV GPS/WiFi applications," in *Proc. Int. Symp. Antennas Propag.*, 2013, vol. 1, pp. 367–370.

[35] C. Cheng, P. Hsiao, H. Kung, and D. Vlah, "Performance measurement of 802.11a wireless links from UAV to ground nodes with various antenna orientations," in *Proc. 15th Int. Conf. Comput. Commun. Netw.*, 2006, pp. 303–308.

[36] K.-H. Loh, B. Golden, and E. Wasil, "Solving the maximum cardinality bin packing problem with a weight annealing-based algorithm," in *Operations Research and Cyber-Infrastructure*. New York, NY, USA: Springer, 2009.

[37] J. Li, G. Kendall, and R. John, "Computing Nash equilibria and evolutionarily stable states of evolutionary games," *IEEE Trans. Evol. Comput.*, vol. 20, no. 3, pp. 460–469, Jun. 2015.

[38] S. Jošilo and G. Dán, "Selfish computation offloading for mobile cloud computing in dense wireless networks," *IEEE Trans. Mobile Comput.*, vol. 18, no. 1, pp. 207–220, Jan. 2019.

[39] S. Sardellitti, G. Scutari, and S. Barbarossa, "Joint optimization of radio and computational resources for multicell mobile-edge computing," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 1, no. 2, pp. 89–103, Jun. 2015.

[40] X. Ma, C. Lin, X. Xiang, and C. Chen, "Game-theoretic analysis of computation offloading for cloudlet-based mobile cloud computing,"in *Proc. 18th ACM Int. Conf. Model. Anal. Simul. Wireless Mobile Syst.*, New York, NY, USA, 2015, pp. 271–278.

[41] A. E. A. A. Abdulla, Z. M. Fadlullah, H. Nishiyama, N. Kato, F. Ono and R. Miura, "Toward fair maximization of energy efficiency in multiple UAS-aided networks: A game-theoretic methodology," *IEEE Trans. Wireless Commun.*, vol. 14, no. 1, pp. 305–316, Jan. 2015.

[42] N. Kumar, S. Misra, N. Chilamkurti, J. H. Lee, and J. J. P. C. Rodrigues, "Bayesian coalition negotiation game as a utility for secure energy management in a vehicles-to-grid environment," *IEEE Trans. Dependable Secure Comput.*, vol. 13, no. 1, pp. 133–145, Jan./Feb. 2016.

[43] G. Huertacanepa and D. Lee, "An adaptable application offloading scheme based on application behavior," in *Proc. 22nd Int. Conf. Adv. Inf. Netw. Appl.Workshops*, 2008, pp. 387–392.

[44] D. Monderer and L. S. Shapley, "Potential games," *Games Econ. Behav.*, vol. 14, no. 1, pp. 124–143, 1996.

[45] M. R. Rahimi, N. Venkatasubramanian, S. Mehrotra, and A. V. Vasilakos, "Mapcloud: Mobile applications on an elastic and scalable 2-tier cloud architecture," in *Proc. IEEE/ACM Int. Conf. Utility Cloud Comput.*, 2012, pp. 83–90.

[46] E. Hyytiä, T. Spyropoulos, and J. Ott, "Offload (only) the right jobs: Robust offloading using the Markov decision processes," in *Proc. IEEE 16th Int. Symp. World Wireless, Mobile Multimedia Netw.*, Jun. 2015, pp. 1–9.

[47] F. Zafari, J. Li, K. K. Leung, D. Towsley, and A. Swami, "Optimal energy consumption with communication, computation, caching and QoI-guarantee," 2018, arXiv:1712.03565.

[48] S. Nazemi, K. K. Leung, and A. Swami, "QoI-aware tradeoff between communication and computation in wireless ad-hoc networks," in *Proc. IEEE 27th Annu. Int. Symp. Pers., Indoor, Mobile Radio Commun.*, Valencia, Spain, 2016, pp. 1–6.

[49] N. Balasubramanian, A. Balasubramanian, and A. Venkataramani, "Energy consumption in mobile phones: A measurement study and implications for network applications," in *Proc. 9th Internet Meas. Conf.*, 2009, pp. 280–293.