Computation Offloading Game for an UAV Network in Mobile Edge Computing

Mohamed-Ayoub Messous[1], Hichem Sedjelmaci[1], Noureddin Houari[2], Sidi-Mohammed Senouci[1]
[1]DRIVE EA1859, Univ. Bourgogne Franche Comté, F58000, Nevers, France.
{ayoub.messous; sid-ahmed-hichem.sedjelmaci; sidi-mohammed.senouci}@u-bourgogne.fr
[2]LSI, Computer Science Department, USTHB, Algiers, Algeria
haouarin@gmail.com

***Abstract*—Due to the limitations of mobile devices in terms of processing power and battery lifetime, cloud based solutions offer an attractive approach to answer these shortcomings. Since offloading intensive computation tasks to an edge/cloud server would achieve impressive performances, computation offloading paradigm has attracted the focus of many research groups in the last few years. This paper considers the problem of computation offloading while achieving a tradeoff between execution time and energy consumption. The proposed solution is intended for a fleet of small drones that are required to achieve highly intensive computation tasks. Drones need to detect, identify and classify objects or situations. Thus, they are brought to deal with intensive tasks such as pattern recognition and video preprocessing. The latter implement very complex calculations and typically require dedicated and powerful processors, which would definitely accentuate the dilemma between energy and delay. We adopted a game theory model where the players are all the drones in the network with three possible strategies. We defined the cost function to be minimized as a combination of energy overhead and delay. The simulation results are very promising and the achieved performances outperformed their counterparts in terms of average system wide cost and scalability.**

Keywords—Computation Offloading; Mobile Edge Computing; Game Theory, UAV (Unmanned Areal Vehicles)

I. INTRODUCTION

Small UAVs (Unmanned Aerial Vehicle), also known as drones, are flying airplanes that do not require a human pilot onboard in order to control its motion. This type of rather small aircrafts can be controlled either remotely or autonomously. In the last decade, UAVs are commonly recognized for their versatile conceivable applications. Despite the fact that relatively large drone platforms are playing increasingly prominent roles in strategic and defense programs, technological advances in the recent years have led to the emergence of smaller significantly cheaper UAVs. This fact made the technology easier to acquire, maintain and handle. As a consequence, their usage was significantly increased and they have known a large expanse in different fields of applications where they proved to be quite useful, not only in military applications, but also in civilian applications such as research and rescue missions, target detection, remote sensing and surveillance tasks [1]. Thus, drones are drought to carry out exploration missions to replace human presence, and/or deliver data to and from areas with no infrastructure [2]. In surveillance applications, UAVs need to detect, classify and identify objects or situations. Thus, they are brought to deal with intensive computation tasks such as pattern recognition algorithms and video sequences preprocessing. These kinds of tasks typically require complex calculations and pattern recognition algorithms, which are known to be extremely intensive computation tasks and therefore require dedicated and powerful processors. Contrariwise, the limited computational capability and low energy resource of UAVs present a major challenge to real-time data processing, networking and decision-making. These requirements are of vital importance to many real-time applications such as forest fire detection and in research and rescue missions. Due to the constrained hardware available on small UAVs, the computation of such computationally intensive tasks may result in slow response times and can also be detrimental to battery life, which may ultimately affect mission success. In order to address the challenges caused by the limited resources and the intermittent connectivity in UAVs network, a cloud-based solution can be adopted [3]. In this context, providing unrestricted computing capabilities at the edge of the access networks is one of the new paradigms that are taking a large magnitude throughout the academic and industrial communities.

In the last few years, Mobile Edge Computing (MEC) was revealed as a very promising concept in order to enhance the network performances, as well as the user experience. Due to the limitations of mobile devices in terms of processing power and battery life, MEC based solutions would offer an attractive approach to undermine these shortcomings. Since intensive computation tasks ought to be offloaded to an edge server, some impressive performances are to be achieved. The existing research works, summarized in [4] and [5], only considered computation offloading to servers in the cloud or in the edge of the access network. The work presented in [6] suggests using a cloudlet based infrastructure in order to further reduce power consumption and network delay while using mobile cloud computing. As computation services can be provided by different systems, this paper considers two possible computation offloading choices: Offload to an edge server

978-1-4673-8999-0/17/$31.00 ©2017 IEEE

through a cellular access network or Offload to a nearby powerful base station via a wireless local network. The processing of intensive tasks will take place in one of these two remote systems and the execution results will be returned eventually to their initiator device, which would significantly reduce power and delay. A new interesting decision problem is defined as: which offloading service to choose, for each computation task, while achieving the best tradeoff between execution time and energy overhead.

In this paper, we formulate a new computation offloading research problem, propose a theoretical game approach for solving the problem, implement a distributed computation offloading algorithm and evaluate the achieved performances. The rest of the paper is organized as follows: we first present the system model in Section II. Section III gives the details of our distributed computation offloading game, the existence of a Nash Equilibrium and the algorithm that goes with. The results achieved through simulation work are presented and discussed in Section IV. Section V discusses some related works and section VI concludes the paper.

II. System Model

This section presents the system model that we have used to implement our computation offloading approach. We consider a set of mobile devices N = {1, 2, ..., N}. Each node executes highly intensive and delay sensitive tasks, while trying to preserve its energy consumption. In order to get useful insights and enable tractable analysis, similar to many previous studies that dealt with mobile cloud computing [7] and mobile edge computing [8] [9], we consider a quasi-static scenario where, during a period of time, the nodes positions remain unchanged, while they may move throughout different periods. The different possibilities for each node are: (i) to perform the task locally, (ii) offload it through a Wi-Fi connection to a neighboring node or a base station, and finally(iii) via a cellular connection to the edge/cloud server. As communication and computation aspects play an important role and affect our approach, we first introduce the communication model for wireless access, then give the details about the computation model in the following subsections.

A. Communication Models

Since cellular access is widely spread, the mobile devices are considered to have a cellular network access along with a second 802.11 wireless interfaces. This latter is used to access the base station while the connection to the edge server is achieved through a cellular network (3G / LTE). We denote $d_n \in \{0, 1, 2\}$ as the computation offloading decision for node n. Explicitly, we have $d_n = 0$ if n chooses to compute its task locally. Otherwise, we would have $d_n = 1$ or $d_n = 2$ if it chooses to offload the computation, respectively, to the edge server via the cellular network or to the base station 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 interferences may incur. It leads subsequently to low data rates, which would negatively affect the performances of the overall network. In a controlled environment and knowing the decisions of all the mobile

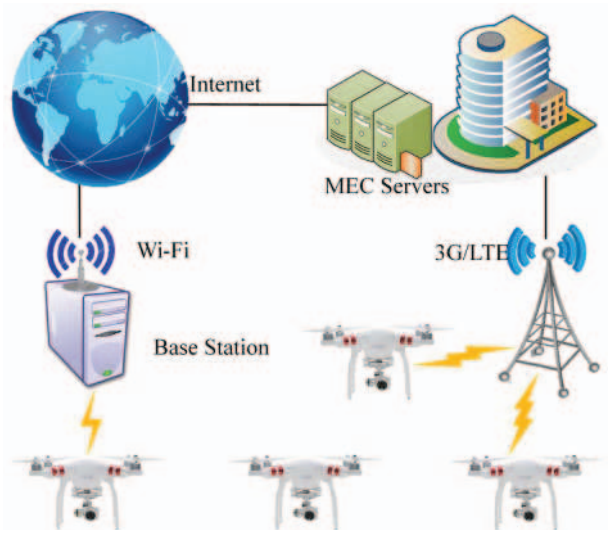

Fig. 1. System Architecture

devices (a_1, a_2, ..., a_N), it is eventually possible to compute the conceivable data rates for each node. Since the scope of the present paper is about the computation offloading problem in its wide form, we chose practical mathematical equations, which use devices coordinates, signal strength and background interferences to compute the actual communication data rate, further details are presented in [10].

B. Computation Model

For the computation aspects of our approach, we consider that each device has one or more computation intensive tasks. Each task i is defined through (Ci, Di) 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 needed 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 a cost function is assumed. The details related to how these cost functions are defined are given in the following paragraphs.

1) Local Computing

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

$$T_{Local} = C_i / F^{Local}_{CPU} \quad (1)$$

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

$$E_{Local} = C_i * e^{Local}_{CPU} \quad (2)$$

Where e^{Local}_{CPU} is the coefficient measure representing the energy consumed per CPU cycle.

2) Offloading to the Edge Server

The first possible offloading approach is to send the computation task via a compatible cellular access network to the edge server. 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 edge server. Therefore, the equation for the time function would be written as:

$$T_{Server} = C_i / F^{Server}_{CPU} + D_i / R_{Cellular} \quad (3)$$

Where F^{Server}_{CPU} represents the frequency of the server CPU, which in practice is very big compared to the running frequency for the mobile devices CPUs. And $R_{Cellular}$ 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 edge server. Thus, the energy function is given as:

$$E_{Server} = D_i * e^{Server}_{Cellular}. \quad (4)$$

Where $e^{Server}_{Cellular}$ denotes the consumption coefficient required to send one unit of data through the cellular network to the edge server.

3) Offloading to theBase Station

The third possible choice and the second offloading approach that we are considering in our work is to offload the computation task through a wireless access point to a nearby base station. 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:

$$T_{BS} = C_i / F^{BS}_{CPU} + D_i / R_{Wi\text{-}Fi} \quad (5)$$

and :

$$E_{BS} = C_i * e^{BS}_{CPU} + D_i * e^{BS}_{Wi\text{-}Fi} \quad (6)$$

Where F^{BS}_{CPU} denotes the CPU's frequency of the base station, $R_{Wi\text{-}Fi}$ is the effective data rate achieved through the wireless local network and e^{BS}_{CPU} measures the energy required to execute one CPU cycle. Finally, $e^{BS}_{Wi\text{-}Fi}$ represents the coefficient measuring the energy needed to send one data unit through the available access point network to the base station.

As many previous studies [8][9][11], 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 the 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.

C. Cost Funtion

The payoff function we defined considers the combination of energy and delay as performance metrics. This is due to the act that the vast majority of mobile devices has limited energy resources. Therefore, the wise management of this critical asset is quite beneficial for the device lifetime. Moreover, intensive computation tasks are known to necessitate a considerable amount of time to complete their execution, even with slightly powerful processors. For these reasons, we implement a global cost function as a joint equation of time delay and energy consumption. The resulting function for each task is given as:

$$Cost = \alpha * T + \beta * E \quad (7)$$

Here α and β represent the weights of delay time and energy respectively. Our aim in using these two parameters is to provide a much higher flexibility and answer a wide range of application specific requirements. Accordingly, depending on the envisioned application or even the actual situation, different tasks can have different weighting parameters. For instance, if the device battery is running low, in order to save more energy, the value of the weight β should be increased. Whereas, for a sensitive to the delay task, the weight $\alpha$is increased in order to reduce the delay.

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

III. Distributed Computation Offloading Strategy

In this section, we consider the issue of achieving the best decentralized decision-making for the offloading of heavy computation tasks either to an edge server or to a neighboring base station. Giving the previously presented computation and communication models, we can behold that the different decisions are mutually related. This means that each device decision will have a direct impact on the other devices throughout the network. If too many nodes choose to offload their intensive computation tasks simultaneously to the edge server or to the same adjacent base station through the same access network, it would have a direct impact on the transmission data rate. Therefore, this may worsen the overhead and subsequently leads to a low network throughput. Moreover, when the data rate is low, much more energy would be consumed to offload the task and it would cause long transmission delays as well. In this case, it would be more efficient to compute the task locally. In order to address the issue of achieving the best possible computation offloading decision, we present in more details throughout the following subsections our theoretical game model.

A. Computation Offloading Game

One of the main reasons for choosing game theory as an enabling framework for our approach is the decentralized nature of the decision making process achieved by the network nodes [12]. Since the latter may have different requirements and

eventually do not pursue the same interest, a decentralized scheme is required. In addition, game theory is perceived as a powerful tool to analyze the interactions among multiple players that are supposed to act toward achieving their own interests. Each player chooses the best possible strategy to achieve its own goals. Thus, by leveraging the intelligence of each individual device, decentralized schemes are devised with low complexity. Consequently, players ought to self-organize into a mutually satisfactory solution such that no player has the incentive to deviate unilaterally. Therefore, it would indeed ease the burden of implementing a more complex centralized system. This mutually satisfactory solution where no player has the incentive to individually change its strategy is called a *Nash Equilibrium* [13].

The game is molded as μ (N, A, G); where N is the number of UAVs/players, A are their strategies and G are their global cost function. a_i is the decided strategy to be carried out by the UAV u_j, which could be either local computing, offload to the edge server or offload to the base station. The strategies of u_j, a_i can be defined as: A $_j$ = {s $_i$; ∀ i ∈ (0 Local computing, 1 Offload to server, 2 Offload to base station)}. u_{-j} other players UAVs excluding player j and a_{-i} their related strategies. G_j (a_i,a_{-i}) is the gain function of UAV u_j; which is the overhead generated by the UAV to carry out one of the strategies cited above; which is defined as shown in Eq. (8).

$$G_j(a_i,a_{-i})=\begin{cases} Z_{Local} = \alpha\, E_{Local} + \beta\, T_{Local}, & if\, s_i = 0 \\ Z_{Server} = \alpha\, E_{Server} + \beta\, T_{Server}, & if\, s_i = 1 \\ Z_{BaseStation} = \alpha\, E_{BS} + \beta\, T_{BS}, & if\, s_i = 2 \end{cases} \quad (8)$$

B. Nash Equilibrium Exsitance Proof

The goal of our game is to determine a consensus between players (u_j, u_{-j}) in order to converge to a Nash Equilibrium (NE) state [14][15][16]. This stable state is defined as an optimal strategy that should be selected by a player$u_j$to get low overhead, while achieving its computation task.

With the help of a potential game [17], we can prove whether our offloading game achieve a NE. Since the potential game has at least one NE solution [9].

Lemma: μ (*N, A, G*) is a potential game

Proof: a game is a potential if its gain function can be expressed as a potential function φ(a) [9][17]. This latter is defined as shown in Eq. (9).

$$\varphi\,(a_i, a_{-i})\text{-}\varphi\,(a'_i, a'_{-i}) = G_j(a_i, a_{-i})\text{-}G_j(a'_i, a'_{-i}) \quad (9)$$

$$\varphi\,(a_i, a_{-i}) = \arg\min_{a_i \in A_j} G_j\,(a_i, a_{-i}) = \psi_i(a_{-i});$$

$$\varphi\,(a'_i, a'_{-i}) = \psi_i(a'_{-i});$$

Where $\psi_i(a_{-i})$ is the best payoff of player u_{-j} given a strategya_{-i}. It's also defined as a *best-response potential game* [17], which is equal to Eq. (10).

$$\psi_i(a_{-i}) = \begin{cases} \arg\min_{a_i \in A_j} Z_{Local}, & if\, s_i = 0 \\ \arg\min_{a_i \in A_j} Z_{Server}, & if\, s_i = 1 \\ \arg\min_{a_i \in A_j} Z_{BaseStation}, & if\, s_i = 2 \end{cases} \quad (10)$$

μ (N, A, G) is a potential game since Eq. (10) satisfies the definition of a potential function and it is the unique 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 $\psi_i(a_{-i})$.In the real experiment the players choose the strategy that corresponds to the minimum of $\arg\min_{a_i \in A_j} Z_{Local}$, $\arg\min_{a_i \in A_j} Z_{Server}$ and $\arg\min_{a_i \in A_j} Z_{BaseStation}$.

C. 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 enables the UAVs to attain a mutually satisfactory goal. The main idea behind *Algorithm 1* is the use of the property of convergence reached thanks to the NE theorem presented in the previous sections. 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 [6] and mobile cloud computing [8] [9]. To implement the concurrently selfish behavior of the different players, we proposed in our case a simple control 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.

Algorithm 1: Distributed Computation Offloading

```
initialization
each UAV selects its first strategy: a_n (0)= 0
compute the initial value of cost function Z0
end initialization
Begin
for each (UAV n ):
    do:
      get the current network status
      select the new best strategy (a i+1)
      compute the new value for Zi+1
        if (Z i< Z i +1) then Z i+1 = Z i
        else send a request to update
      if (request accepted) then update strategy (a i)
      else Z i+1= Z i
    until an Equilibrium is achieved
end for
End
```

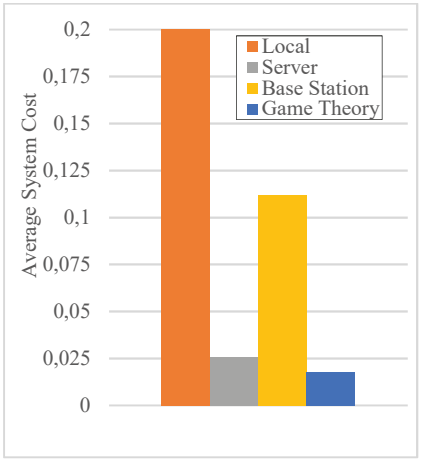

Fig. 2. Average System wide cost

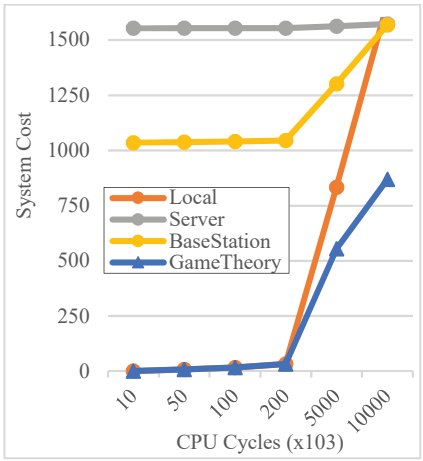

Fig. 3. System cost with different CPU cycles

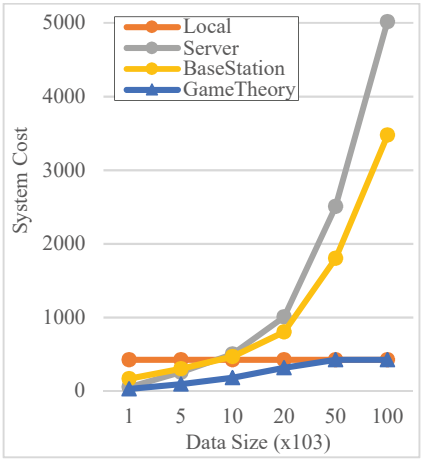

Fig. 4. System cost with different Data sizes

IV. EXPERIMENTAL RESULTS AND DISCUSSION

We evaluate, in this section, the performances of our game theory based approach compared to three other strategies: (i) Local Computing, (ii) Offloading to Server, and (iii) Offloading to Base Station. In the first model, all the computation tasks are executed locally. However, they are offloaded to edge server via a cellular access network in the second approach and to the base station via a wireless local network in the third approach. We evaluated the system wide cost function, presented previously in section II.C. We considered that energy and delay are equally important thus we chose $\alpha = \beta = ½$. We evaluate scenarios with different number of UAVs, from 5 up till 50. We study also the impact that data size and computation cycles of the computation tasks have on the overall system cost. The main simulation parameters are summarized in Table 1.

We consider the CPU capability of the edge server $F^{Server}{}_{CPU}$ to be ten times more powerful than base station 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 consumption coefficients, we selected realistic measures. We consider that sending one data unit through wireless local area interface $e^{BS}{}_{Wi\text{-}Fi}$ consumes one thousand time more than computing one CPU cycle locally $e^{Local}{}_{CPU}$. Which is twice the energy consumed if the calculation is executed in the base station $e^{BS}{}_{CPU}$. As for the energy coefficient of the server; we consider it unlimited, thus $e^{Server}{}_{CPU} = 0$. And for $e^{Server}{}_{Cellular}$, we consider it 20% less efficient than $e^{BS}{}_{Wi\text{-}Fi}$.

TABLE I. SIMULATION PARAMETERS

Scenarios	
UAVs #	[5, ... 50]
Tasks: C_i	[10, … 1000] ($x10^3$)
Tasks: D_i	[1, … 100] ($x10^3$)
(α , β)	(½ , ½)
Parameters	Values
$F^{Local}{}_{CPU}$	*1 GHz*
$F^{Server}{}_{CPU}$	*50 GHz*
$F^{BS}{}_{CPU}$	*5 GHz*
$e^{Local}{}_{CPU}$	*1 u*
$e^{Server}{}_{CPU}$	*0 u*
$e^{BS}{}_{CPU}$	*½ u*
$R_{Cellular}$	*1 Mbps*
$R_{Wi\text{-}Fi}$	*5 Mbps*
$e^{Server}{}_{Cellular}$	*1200 u*
$e^{BS}{}_{Wi\text{-}Fi}$	*1000 u*

The diagram shown in figure 2 reveals that our approach outperforms the three models in terms of average system wide cost. This is due to the fact that our model always chooses the most efficient strategy to lower time overhead and energy consumption.

To investigate the impact that computation cycles have on our approach, we performed simulation while fixing the number of CPU cycles Ci each time and changing the size of data. The results are shown in Figure 3. We see that the average system cost achieved through our game theory based approach outperforms the three other models in all the cases. The achieved results were even better for very intensive computation tasks. The system cost for the game theory based model increases much slower because as the number of processing cycles increases, more UAVs choose to offload their tasks to mitigate the heavy cost of local computing. Figure 3 shows also that the local computing is most suitable for less intensive computation tasks. Inversely, offloading to server is not appropriate for less intensive tasks.

To evaluate the impact of data size of computation tasks, we finally run simulations with different data sizes while changing each time the computation cycles and the number of UAVs. Figure 4 shows that the system cost increases as the data size increases in the two offloading approaches, due to the fact that big data induces high transmission overhead. Nevertheless, system cost 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.

V. RELATED WORKS

Computation offloading to the cloud or to an edge server can significantly increase the computational capability of

constrained devices, but the response times may suffer when many devices attempt to offload their computation tasks simultaneously. This may primarily be due to the concurrently access to a possibly constrained network resources [8]. A significant body of works focuses on the problem of computation offloading in order to improve the time response and to minimize energy consumption. The experimental analysis led in [18] confirmed that the remote execution of computation tasks could achieve substantial battery power savings. The authors in [19] proposed a cooperative offloading model while achieving a tradeoff between energy and network traffic in a wireless local area network. Their model tried to reduce the computation repetition and eliminate traffic redundancy via cooperative execution of tasks and sharing computation results. [20] presented the offloading problem as a Markov decision process and claim to provide a near-optimal offloading strategy. They adopted a stochastic model in order to study the impact of dynamic arrival of offloading tasks. They incorporated also in their model occasionally available wireless access links. Unlike this work, we consider in our design an always accessible network. [21] introduces a Cloudlet based mobile cloud computing system aiming to reduce the power consumption and the network delay while using mobile cloud access.

Some recent work used game theory as an enabling tool for their design of computation offloading approaches [8][9]. Authors in [8] proposed a single wireless channel to access an all-powerful cloud. They proved the game to be potential game and the players are the mobile devices with two possible strategies. Then they extended their model in [9] to a more general use case with multiple wireless channels and shows that the game still is a potential game. On the contrary to these works, we consider in our approach a fair equitable bandwidth sharing. Besides describing an all new scheme of the computation offloading problem for a UAV network based on game theory, we design a non-cooperative theoretical game with *N* player, each of which has three different strategies. We prove the existence of a Nash equilibrium where no player has the incentive to deviate from.

VI. Conclusion

We considered throughout this paper the problem of offloading highly intensive computation tasks in order to improve the energy overhead and decrease execution delay. We have proposed a theoretical game approach where the players are a set of UAVs being led to carry out heavy computation tasks. Three possible strategies are defined: (i) compute the task locally, (ii) offload the task via cellular network to a server in the edge, and finally (iii) offload the task to a base station via a wireless local network. We defined a Nash Equilibrium and proved its existence. Then, we implemented our offloading approach through a distributed algorithm and extensively tested the different performances in simulation work. The cost function is defined as a combination between energy and delay. The proposed approach provides good performances and reduces the average system wide cost. In addition, the obtained results outperform other approaches in terms of scaling mode.

References

[1] G. Pajares, “Overview and Current Status of Remote Sensing Applications Based on Unmanned Aerial Vehicles (UAVs),” Photogramm. Eng. Remote Sens., vol. 81, no. April, pp. 281–330, 2015.

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

[3] C. Luo, J. Nightingale, and C. Grecos, “A UAV-Cloud System for Disaster Sensing Applications,” 2015.

[4] A. Ahmed and E. Ahmed, “A Survey on Mobile Edge Computing,” 10th IEEE International Conference on Intelligent Systems and Control, (ISCO 2016), no. JANUARY, 2016.

[5] M. T. Beck, M. Werner, S. Feld, and T. Schimper, “Mobile Edge Computing : A Taxonomy,” in The Sixth International Conference on Advances in Future Internet (AFIN 2014), 2014, no. c, pp. 48–54.

[6] Y. Jararweh, L. Tawalbeh, F. Ababneh, and F. Dosari, “Resource efficient mobile computing using cloudlet infrastructure,” Proc. - IEEE 9th Int. Conf. Mob. Ad-Hoc Sens. Networks, MSN 2013, pp. 373–377, 2013.

[7] Xiao Ma, Chuang Lin, Xudong Xiang, and Congjie Chen. 2015. Game-theoretic Analysis of Computation Offloading for Cloudlet-based Mobile Cloud Computing. In Proceedings of the 18th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM '15). ACM, New York, NY, USA, 271-278.

[8] X. Chen, “Decentralized Computation Offloading Game For Mobile Cloud Computing”,IEEE Transactions on Parallel and Distributed Systems, vol. 26, no. 4, pp. 974–983, 2015..

[9] X. Chen, L. Jiao, W. Li, and X. Fu, “Efficient Multi-User Computation Offloading for Mobile-Edge Cloud Computing,” *IEEE/ACM Trans. Netw.*, 2015.

[10] Theodore Rappaport. 2001. Wireless Communications: Principles and Practice (2nd ed.). Prentice Hall PTR, Upper Saddle River, NJ, USA.

[11] D. Huang, P. Wang, and D. Niyato, “A dynamic offloading algorithm for mobile computing,” IEEE Transactions on Wireless Communications, vol. 11, no. 6, pp. 1991–1995, 2012.

[12] M. Attia, H. Sedjelmaci, SM. Senouci and E. Aglzim, "Game Model to Optimally Combine Electric Vehicles with Green and Non-Green Sources into an End-to-End Smart Grid Architecture", Journal of Networking and Computer Applications,Vol 72, 2016,pp 1-13.

[13] J.Li, G. Kendall, R.John, “Computing nash equilibria and evolutionarily stable states of evolutionary games”, *IEEE Transactions on Evolutionary Computation*, Vol 20, Issue 3, 2015.

[14] A. E. A. A. Abdulla, Z. M. Fadlullah, H. Nishiyama, N. Kato, F. Ono, and R. Miura, “Towards 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.

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

[16] H. Sedjelmaci, S M Senouci, N. Ansari, “Intrusion Detection and Ejection Framework Against Lethal Attacks in UAV-Aided Networks: A Bayesian Game-Theoretic Methodology; Accepted on *IEEE Transactions on Intelligent Transportation Systems*, 2016.

[17] D. Monderer, LS. Shapley, “Potential games”, *Games and Economic Behavior*. vol. 14, no. 1, pp. 124–143, 1996.

[18] A. Rudenko, P. Reiher, G. J. Popek, and G. H. Kuenning, “Saving portable computer battery power through remote process execu- tion,” ACM Mob. Comput. Commun. Rev., vol. 2, no. 1, pp. 19–26, Jan 1998

[19] J. Song, Y. Cui, M. Li, J. Qiu, and R. Buyya, “Energy-traffic tradeoff cooperative offloading for mobile cloud computing,” IEEE Int. Work. Qual. Serv. IWQoS, pp. 284–289, 2014.

[20] E. Hyytiä, T. Spyropoulos, and J. Ott, “Offload (only) the right jobs: Robust offloading using the Markov decision processes,” in Proc. of IEEE WoWMoM, Jun. 2015, pp. 1–9.

[21] Y. Jararweh, L. Tawalbeh, F. Ababneh, and F. Dosari, “Resource efficient mobile computing using cloudlet infrastructure,” Proc. - IEEE 9th Int. Conf. Mob. Ad-Hoc Sens. Networks, MSN 2013, pp. 373–377, 2013