future internet

MDPI

Article

Intelligent Dynamic Data Offloading in a Competitive Mobile Edge Computing Market

Giorgos Mitsis [1], Pavlos Athanasios Apostolopoulos [2], Eirini Eleni Tsiropoulou [2,*] and Symeon Papavassiliou [1]

[1] School of Electrical and Computer Engineering, National Technical University of Athens, 15780 Athina, Greece; gmitsis@netmode.ntua.gr (G.M.); papavass@mail.ntua.gr (S.P.)
[2] Department of Electrical and Computer Engineering, University of New Mexico, Albuquerque, NM 87131, USA; pavlosapost@unm.edu
* Correspondence: eirini@unm.edu

Received: 12 April 2019; Accepted: 13 May 2019; Published: 21 May 2019

Abstract: Software Defined Networks (SDN) and Mobile Edge Computing (MEC), capable of dynamically managing and satisfying the end-users computing demands, have emerged as key enabling technologies of 5G networks. In this paper, the joint problem of MEC server selection by the end-users and their optimal data offloading, as well as the optimal price setting by the MEC servers is studied in a multiple MEC servers and multiple end-users environment. The flexibility and programmability offered by the SDN technology enables the realistic implementation of the proposed framework. Initially, an SDN controller executes a reinforcement learning framework based on the theory of stochastic learning automata towards enabling the end-users to select a MEC server to offload their data. The discount offered by the MEC server, its congestion and its penetration in terms of serving end-users' computing tasks, and its announced pricing for its computing services are considered in the overall MEC selection process. To determine the end-users' data offloading portion to the selected MEC server, a non-cooperative game among the end-users of each server is formulated and the existence and uniqueness of the corresponding Nash Equilibrium is shown. An optimization problem of maximizing the MEC servers' profit is formulated and solved to determine the MEC servers' optimal pricing with respect to their offered computing services and the received offloaded data. To realize the proposed framework, an iterative and low-complexity algorithm is introduced and designed. The performance of the proposed approach was evaluated through modeling and simulation under several scenarios, with both homogeneous and heterogeneous end-users.

Keywords: software defined networks; mobile edge computing; reinforcement learning; stochastic learning automata; game theory; data offloading; pricing; optimization

1. Introduction

Mobile Edge Computing (MEC) has emerged as a vital solution to offer computing resources at the edge of the network and in close vicinity to the mobile end-users. The end-users are able to offload their computation tasks to the MEC servers, which can further process the subscribers' offloaded tasks. The concept of MEC was motivated by the unprecedented growth of mobile traffic, especially by the smart phones, and the emergence of enhanced multimedia services, which are characterized by high computing demands. The main benefits of the MEC technology are: its potential to reduce the latency, provide location-awareness, improve the performance of the mobile applications, reduce the energy consumption of the mobile devices by alleviating the burden of executing their computing tasks locally, and provide accurate computing outcomes in a time-wise manner. However, the adoption of the MEC technology in the overall networking architecture has created the need of devising control

mechanisms to route the mobile end-users offloading tasks to the MEC servers, while accounting for network's congestion, MEC servers computation capabilities and end-users Quality of Service (QoS) prerequisites. Towards this direction, Software Defined Networking (SDN) is another 5G enabling technology complementing the MEC advancement towards designing dynamic, manageable, adaptable, and cost-effective networks. Specifically, the SDN paradigm transforms the communications networks into a programmable world, where a centralized entity, i.e., SDN controller, has a global view of the communication links and manages the network traffic more efficiently and dynamically. The MEC environment can substantially benefit from the SDN technology, as the decision making with respect to end-users' selection of the specific MEC server to perform their data offloading, the routing of the end-users' offloading traffic and the guarantee of the end-users' QoS constraints can be performed in the control plane, which is implemented within the SDN controller, in a dynamic manner.

1.1. Related Work

The problem of data offloading from the end-users to the MEC servers for further computing has been extensively studied in the recent literature, while examining both the computation and the communication limitations [1]. In [2], a minimization problem of the long-term average weighted total devices' and MEC server's power consumption is formulated and solved in a multi-user MEC environment, concluding to a joint radio and computing resource management scheme, where both the optimal users' transmission power to offload their data and the corresponding computing power to process them are determined. In [3], a femto-based MEC environment is introduced and the authors exploit the trade-off between the end-users' energy consumption and latency towards minimizing the end-users' affordable latency while executing an application. A centralized optimization problem is introduced in [4] targeting at the minimization of the weighted sum end-users' energy consumption, while accounting for the end-users' computation latency constraints. The authors consider that the end-users adopt the orthogonal frequency division multiple access technique to offload their data to the MEC servers and they capture the end-users' workload offloading priorities in the problem formulation and solution, while a similar approach is also followed in [5]. In [6], the authors proposed a joint resource allocation scheme of the computation and communication resources of the MEC system aiming to minimize the end-users' energy consumption and the latency of the applications' execution at the MEC servers. Moreover, in [7], the authors focused on the energy efficient operation of the MEC system and they propose a dynamic data offloading and resource allocation scheme to minimize the computation application completion time and the end-users' energy consumption. A holistic framework of minimizing the total cost of energy, computation, and delay for the end-users is introduced in [8].

Game Theory has also been adopted to deal with the data offloading problem in the MEC environment, while providing the enhanced flexibility to the end-users to make autonomous data offloading decisions in a distributed manner [9]. In [10], a data offloading decision-making game is formulated among the end-users, who decide the amount of data that will be offloaded to a single MEC server, as well as the part of the computation task that will be executed locally at their devices. A similar problem is addressed in [11], while a multiple MEC servers environment is considered and the end-users have to additionally select to which MEC server they will offload part of their data. The problem of activating the MEC servers based on the end-users computing demands is addressed in [12], where the MEC servers' activation problem is formulated as a minority game and a distributed reinforcement learning algorithm is executed by each MEC server in order to determine if it will be active or not. The concept of applying usage-based pricing policies to the end-users while they exploit the MEC servers' computing capabilities is introduced in [13,14] towards providing incentives to the end-users to consume the MEC servers' computing services in a fair manner.

Recently, the capabilities of the SDN have been exploited by the MEC environment to efficiently and effectively deal with the data offloading problem, the activation of the MEC servers, the routing of the end-users offloading data, and the announcement of pricing mechanisms to control the smooth

operation of the MEC system [15]. The problem of selecting a computing mode (i.e., local, MEC, or cloud computing) for each end-users' computation task is studied in [16], where the SDN controller executes the Computing Mode Selection algorithm and announces the corresponding routing policies to the end-users. The benefits of the combined use of SDN and MEC within the Internet of Things (IoT) systems are discussed in detail in the surveys [17,18]. In [19], a smart e-health IoT service is introduced, which is based on SDN-powered MEC within a vehicular ad-hoc network architecture to detect heart attacks in a real-time manner. In [20], the authors focused on virtual reality and vehicular IoT applications and they propose an SDN-based MEC framework to provide the necessary data-plane flexibility, programmability, and reduced latency. Furthermore, in [21], the adoption of SDN and MEC is presented to overcome the barriers of network densification of IoT cloud integration within a smart home environment.

Following the above discussion, the interest of the research community in the SDN-based MEC framework is growing and calls for advanced flexible, dynamic and programmable data offloading mechanisms. In addition, the problem of monetary-based pricing of the MEC servers computing services has not been addressed in the existing literature.

1.2. Contributions and Outline

This paper aims at filling the aforementioned research gaps by proposing an SDN-powered MEC architecture towards dealing with the joint problem of intelligent MEC server selection and end-users' data offloading in a multiple MEC server and multiple end-user environment. From an architectural point of view, the joint optimization problem is solved at the SDN controller, which announces the best strategies to the MEC servers and the end-users. The goal of the MEC servers is to maximize their profit by serving the end-users' computing demands, while the end-users aim at maximizing their perceived satisfaction (expressed through representative utility functions) by the provided computing services from the MEC environment.

The key contributions of our work that differentiate it from the rest of the literature are summarized as follows.

1. The monetary-based pricing of the MEC servers' computing services, the offered discount to the end-users, the total end-users' offloaded workload, and the cost of the MEC servers to process the workload are considered towards formulating representative welfare functions for the MEC servers, creating a multi-server competitive computing market. In addition, the end-users' perceived satisfaction from executing their tasks to the MEC servers is captured in holistic utility functions, while considering the corresponding cost that the end-users have to pay in order to enjoy the requested services (Section 2).
2. A reinforcement learning framework is included within the SDN controller's functionalities towards implementing the MEC server selection by the end-users to offload their data for further processing. The theory of stochastic learning automata is adopted, where the end-users are represented as stochastic agents at the SDN controller, learning the best MEC server selection based on their previous actions and the reaction of the MEC environment. Each MEC server is characterized by a reputation score, which acts as a reward probability to the MEC server selection process. The reputation score captures the discount offered by the MEC server, its congestion in terms of serving end-users' computing tasks, its penetration in terms of serving the end-users' computing demands, and its announced pricing for its computing services (Section 3).
3. Following the reinforcement learning-based MEC server selection by the end-users, a two-layer optimization problem is formulated and solved aiming at maximizing the MEC servers' profit and also maximizing the perceived satisfaction by the end-users. The end-users' maximization problem of their satisfaction is addressed at the first stage as a non-cooperative game among the end-users, who practically aim at maximizing their personal utility function. A Nash Equilibrium (NE) point is determined, which expresses the optimal amount of offloading data for each end-user. At the second stage of the joint optimization problem, given the end-users' offloaded

data, an optimization problem of each MEC server's profit (as expressed by its welfare function) is formulated and solved by the MEC servers (Section 4).

4. An iterative and low-complexity algorithm is introduced to implement the MEC server selection process based on reinforcement learning and determine the optimal MEC servers' computing services' monetary pricing and end-users' optimal data offloading based on game-theoretic and optimization techniques (Section 5).
5. A series of detailed simulation experiments are performed to evaluate the performance and inherent attributes of the proposed framework. Furthermore, a comparative study demonstrates its superiority and benefits compared to other relevant alternative strategies (Section 6).

Finally, Section 7 concludes the paper.

2. SDN-Powered Mobile Edge Computing

The SDN-powered MEC architecture consisting of multiple MEC servers and multiple end-users is presented in Figure 1. Each MEC server $s, s \in S$, $S = [1, ..., s, ...|S|]$ communicates with the SDN controller towards setting the price $p_s^{(t)}[\$/bits]$ of its computing services per time slot t. The whole operation of the examined system is divided in time slots, where $T = [1, ...t, ..., |T|]$ denotes their corresponding set. At each time slot, the SDN controller determines the MEC server selection by the end-users (see Section 3), as well as the optimal price $p_s^{(t)}$ for each MEC server and the optimal data offloading $b_{u,s}^{(t)}[bits]$ of each end-user u to the selected server s (see Section 4). Each end-user $u, u \in U, U = [1, ..., u, ...|U|]$ receives the required information by the SDN controller to offload its data $b_{u,s}^{(t)}$ to the selected server s. Each end-user u has a maximum amount of data $I_u^{(t)}$ that should be processed to perform a computing task, and part of them are offloaded to the MEC server, i.e., $b_{u,s}^{(t)} \in A_u^{(t)} = [0, I_u^{(t)}]$, while the rest of the data are processed locally.

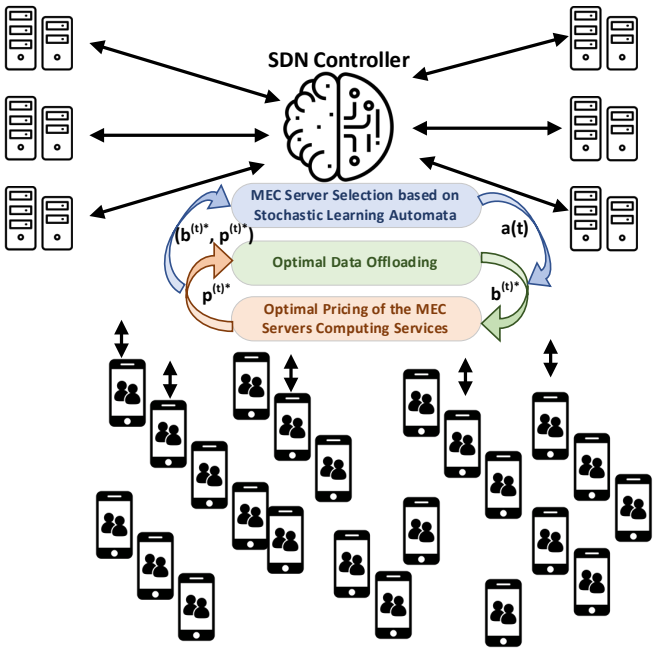


Figure 1. SDN-powered MEC architecture.

2.1. End-User Utility Function

At the beginning of each time slot, each end-user u sends to the SDN controller its total computing demands $I_u^{(t)}$ that are needed to execute a computing task, while the SDN controller determines the optimal amount of offloaded data $b_{u,s}^{(t)}$ for end-user u at the MEC server s, as explained in detail in

Section 4. Given that the MEC servers have bounded and limited computing capabilities, the data offloading strategies of the rest of the end-users, i.e., $\mathbf{b}_{-\mathbf{u}}^{(\mathbf{t})}$, contribute to the configuration of the prices announced by the MEC servers and influence the data offloading $b_{u,s}^{(t)}$ of end-user u. Thus, towards formulating the user's u perceived satisfaction, the end-user's relative data offloading is defined as $r_u^{(t)} = \frac{b_{u,s}^{(t)}}{B_{-u}^{(t)}}$, where $B_{-u}^{(t)} = \sum_{s \in S} \sum_{u' \in U, u' \neq u} b_{u',s}^{(t)}$ expresses the total data offloading of the rest of the end-users $u', u' \in U - \{u\}$. The end-user's actual perceived satisfaction $s_u^{(t)}$ at time slot t is increasing with respect to its relative data offloading $b_{u,s}^{(t)}$, as part of the requested computing task is offloaded to the MEC server and does not consume the end-user's local computing resources. In addition, after the end-user offloads its total data $I_u^{(t)}$ to the MEC server, its perceived satisfaction is saturated as the end-user cannot benefit more by the MEC server's computing services, as presented in Figure 2. Thus, without loss of generality and for presentation purposes only, in this paper, we adopt a logarithmic function with respect to the end-user's offloaded data $b_{u,s}^{(t)}$ to capture end-user's actual perceived satisfaction, as follows.

$$s_u^{(t)}(b_{u,s}^{(t)}, \mathbf{b}_{-\mathbf{u}}^{(\mathbf{t})}) = \alpha_u log(1 + \beta_u r_u^{(t)}) \tag{1}$$

where $\mathbf{b}_{-\mathbf{u}}^{(\mathbf{t})}$ is the vector of all end-users data offloading excluding end-user u, and the $\alpha_u, \beta_u \in \mathbb{R}^+$ parameters determine the slope of the logarithmic function in a personalized manner for end-user u, thus expressing how easily or not an end-user u becomes satisfied by offloading its data to the MEC server.

Additionally, each end-user is charged for using the MEC server's computing services in a fair manner according to its relative data offloading. This policy enables even the low-budget end-users to exploit the MEC servers' capabilities to some degree, by prohibiting the high-budget ones from dominating the system. Thus, the cost function of end-user's u offloaded data is formulated as follows.

$$c_u^{(t)}(b_{u,s}^{(t)}, \mathbf{b}_{-\mathbf{u}}^{(\mathbf{t})}) = d_u^{(t)} p_s^{(t)} r_u^{(t)} \tag{2}$$

where $d_u^{(t)}, d_u^{(t)} \in \mathbb{R}^+$ expresses end-user's u spending dynamics in order to use the MEC server's computing services. Specifically, a smaller value of $d_u^{(t)}$ reflects end-user's u dynamic behavior to spend more money in order to buy computing support from the MEC servers. The price announced by the MEC server s is denoted as $p_s^{(t)}[\$/bits]$.

Following the above analysis, end-user's u utility function captures both its actual perceived satisfaction $s_u^{(t)}$ and its corresponding cost $c_u^{(t)}$ in order to enjoy the MEC server's computing services. The end-user's u utility function is defined as follows.

$$U_u^{(t)}(b_{u,s}^{(t)}, \mathbf{b}_{-\mathbf{u}}^{(\mathbf{t})}, \mathbf{p}^{(\mathbf{t})}) = s_u^{(t)}(b_{u,s}^{(t)}, \mathbf{b}_{-\mathbf{u}}^{(\mathbf{t})}) - c_u^{(t)}(b_{u,s}^{(t)}, \mathbf{b}_{-\mathbf{u}}^{(\mathbf{t})}) = \alpha_u log(1 + \beta_u r_u^{(t)}) - d_u^{(t)} p_s^{(t)} r_u^{(t)} \tag{3}$$

where $\mathbf{p}^{(\mathbf{t})} = [p_1^{(t)}, ..., p_s^{(t)}, ..., p_{|S|}^{(t)}]$ denotes the vector of the prices announced by all the MEC servers.

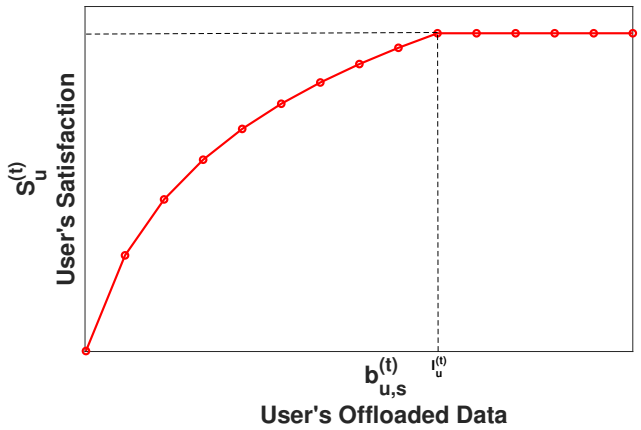


Figure 2. End-user's actual perceived satisfaction by data offloading.

2.2. Mobile Edge Computing Server Characteristics and Profit

Each MEC server s supports a total computing demand of the end-users per time slot equal to $\sum_{u\in U} b_{u,s}^{(t)}$ from all the end-users that selected the specific MEC server to offload their data. In addition, towards providing incentives to the end-users to select a specific MEC server to be served from, the latter provides some discounts $f_s^{(t)}$ expressed as a percentage of the original announced price of its computing services. Furthermore, the MEC server has an actual cost $c_s^{(t)}[\$/bits]$ in order to process the amount of data that it receives. Please also be reminded that the MEC server charges the end-users with a price $p_s^{(t)}[\$/bits]$ for the computing services that it offers.

Additionally, a MEC server increases its positive reputation towards the end-users if it is characterized by a good penetration within the end-users' computing demands. Specifically, the penetration of a MEC server s is defined as the total amount of data that the server s processed over the total amount of data that are processed within the SDN-powered MEC system for a total time period T, i.e., $\frac{\sum_{t\in\{1,\ldots,T\}}\sum_{u\in U} b_{u,s}^{(t)}}{\sum_{s\in S}\sum_{t\in\{1,\ldots,T\}}\sum_{u\in U} b_{u,s}^{(t)}}$. In addition, we assume that each MEC server s can handle a total amount of data B_s^{Max}. Thus, an indicative parameter showing the congestion of the MEC server per time slot in terms of processing the end-users' offloaded data is expressed as the ratio of the total amount of data $\sum_{u\in U} b_{u,s}^{(t)}$ that the MEC server processes in time slot t over its total computing capability of data B_s^{Max}, i.e., $CONG_s = \frac{\sum_{u\in U} b_{u,s}^{(t)}}{B_s^{Max}}$.

Following the above analysis and combining all the aforementioned factors and parameters that characterize the MEC server s, its reputation score within the SDN-powered MEC environment is defined as follows.

$$R_s^{(t)} = w_1 \frac{\frac{\sum_{k\neq s}[(1-f_k^{(t)})]p_k^{(t)}}{K}}{(1-f_s^{(t)})p_s^{(t)}} + w_2 \frac{1}{(1+CONG_s)^3} + w_3 \frac{\sum_{t\in\{1,\ldots,T\}}\sum_{u\in U} b_{u,s}^{(t)}}{\sum_{s\in S}\sum_{t\in\{1,\ldots,T\}}\sum_{u\in U} b_{u,s}^{(t)}} \quad (4)$$

In Equation (4), the first term expresses the relative pricing of a MEC server s in terms of offering its computing services to the end-users, the second term expresses the level of MEC server's congestion towards serving the end-users, while the third term expresses its penetration in serving end-users' computing demands. The weights w_1, w_2, w_3 are configurable parameters that express the relative weight of each term within our study, and it should hold true that $w_1 + w_2 + w_3 = 1$.

The revenue of each MEC server s from processing a total amount of end-users' offloaded data $\sum_{u\in U} b_{u,s}^{(t)}$ depends on the announced price $p_s^{(t)}$ and the corresponding discount $f_s^{(t)}$ that the MEC server provides, and is given as follows.

$$REV_s^{(t)}(\mathbf{b^{(t)}}, \mathbf{p^{(t)}}) = (1-f_s^{(t)})p_s^{(t)} \sum_{u\in U} b_{u,s}^{(t)} \quad (5)$$

where $\mathbf{b^{(t)}}$ is the data offloading vector of all the end-users and $\mathbf{p^{(t)}}$ denotes the announced prices by all the MEC servers in the system.

On the other hand, the MEC server's total monetary cost to perform the processing of the offloaded data, is given as follows.

$$C_s^{(t)}(\mathbf{b^{(t)}}) = c_s^{(t)} \sum_{u \in U} b_{u,s}^{(t)} \tag{6}$$

where $c_s^{(t)}$ is the MEC server's s computing cost per unit of data. Thus, the MEC server's profit is concluded by subtracting its cost from its revenue and is given as follows.

$$P_s^{(t)}(\mathbf{b^{(t)}}, \mathbf{p^{(t)}}) = REV_s^{(t)}(\mathbf{b^{(t)}}, \mathbf{p^{(t)}}) - C_s^{(t)}(\mathbf{b^{(t)}}) = (1 - f_s^{(t)}) p_s^{(t)} \sum_{u \in U} b_{u,s}^{(t)} - c_s^{(t)} \sum_{u \in U} b_{u,s}^{(t)} \tag{7}$$

3. MEC as a Learning System

At the SDN controller's side, the end-users are represented and considered as stochastic learning automata that sense the environment and make future decisions based on their past experience. At each time slot t, the end-user can select to be served by a MEC server s, thus, the set of end-users' actions at time slot t is $a(t) = \{a_1, ..., a_s, ..., a_S\}$. The SDN controller has the information of the end-users' offloaded data $\mathbf{b^{(t)}}$ and the prices $\mathbf{p^{(t)}}$ that the MEC servers announce regarding offering their computing services. The SDN controller can determine the reputation score $R_s^{(t)}$ for each MEC server, which can be normalized towards defining the reward probability as follows.

$$rew_s^{(t)} = \frac{R_s^{(t)}}{\sum_{s \in S} R_s^{(t)}} \tag{8}$$

The reward probability $rew_s^{(t)}, 0 \leq rew_s^{(t)} \leq 1$ represents the potential reward that an end-user may experience by choosing to offload its data to the MEC server s. Following the theory of the stochastic learning automata, the action probability vector of an end-user $u, u \in U$ is $\mathbf{Pr_u^{(t)}} = [Pr_{u,1}^{(t)}, ..., Pr_{u,s}^{(t)}, ..., Pr_{u,S}^{(t)}]$, where $Pr_{u,s}^{(t)}$ is defined as the probability of the end-user u to select the MEC server s to offload its data. Based on the theory of stochastic learning automata [22], the rule of updating the end-users' action probabilities at the SDN controller is defined as follows.

$$Pr_{u,s}^{(t+1)} = Pr_{u,s}^{(t)} - b \cdot rew_s^{(t)} \cdot Pr_{u,s}^{(t)}, \quad s^{(t+1)} \neq s^{(t)} \tag{9a}$$

$$Pr_{u,s}^{(t+1)} = Pr_{u,s}^{(t)} + b \cdot rew_s^{(t)} \cdot (1 - Pr_{u,s}^{(t)}), \quad s^{(t+1)} = s^{(t)} \tag{9b}$$

where $0 < b < 1$ denotes the learning parameter expressing how fast the end-users explore the available options of the MEC servers towards offloading their data. Equation (9a) represents the probability of end-user u selecting a different MEC server to offload its data in the next time slot $t+1$ compared to end-user's choice in the current time slot t, while Equation (9b) expresses the probability of end-user u to keep being served by the same MEC server. It is noted that, initially, the end-users' action probabilities are initialized as $Pr_{u,s}^{(t=0)} = \frac{1}{S}$. The MEC servers selection learning process executed at the SDN controller is presented in the Data Offloading and MEC Server Selection (DO-MECS) algorithm (see Section 5).

4. Autonomous Data Offloading and Price Setting

4.1. Problem Formulation

Following the above described reinforcement learning technique of the stochastic learning automata, each end-user has concluded to the selection of a MEC server to offload its data. Then, the goal of each MEC server is to maximize its profit by processing the end-users' data, while the

goal of each end-user is to maximize its perceived satisfaction, as expressed by its utility function, by offloading the optimal amount of data to the selected MEC server. Thus, a two-layer optimization problem is formulated, as follows.

$$\mathbf{b^{(t)}}^* = argmax_{b_{u,s}^{(t)}} U_u^{(t)}(b_{u,s}^{(t)}, \mathbf{b_{-u}^{(t)}}, \mathbf{p^{(t)}}) \tag{10a}$$

$$\mathbf{p^{(t)}}^* = argmax_{\mathbf{p^{(t)}}} P_s^{(t)}(\mathbf{b^{(t)}}, \mathbf{p^{(t)}}) \tag{10b}$$

As it is observed by Equations (10a) and (10b), the MEC servers optimal price $\mathbf{p^{(t)}}^*$ and the end-users optimal data offloading $\mathbf{b^{(t)}}^*$ are interdependent, thus the joint optimization problem is formulated as a two-layer optimization framework. Initially, the end-users determine their optimal data offloading $\mathbf{b^{(t)}}^*$ via confronting the optimization problem of their personal utility functions as a non-cooperative game among them. Then, at the second layer, the MEC servers determine their optimal announced prices $\mathbf{p^{(t)}}^*$ given the data offloading of the end-users, via solving an optimization problem. The formulation and solution of the optimization problem is performed at the SDN controller, where its advanced computing capabilities enable the fast decision-making. In the following two subsections, we analyze in detail each layer of the optimization problem.

4.2. Optimal Data Offloading

At first, the optimal data offloading $b_{u,s}^{(t)*}$ of each end-user u that has selected to offload its data to the MEC server s at the time slot t is determined. A non-cooperative game $G = [U, \{A_u^{(t)}\}, \{U_u^{(t)}\}]$ is formulated among the end-users who compete with each other towards determining their optimal data offloading. The game G consists of three components: (a) the set of end-users (i.e., players) $U = [1, ..., u, ..., |U|]$;; (b) the strategy space $A_u^{(t)} = [0, I_u^{(t)}]$, where $b_{u,s}^{(t)} \in A_u^{(t)}$; and (c) the end-user's utility function $U_u^{(t)}$. Each end-user wants to maximize its personal utility function, while considering the physical limitations, as follows.

$$max_{b_{u,s}^{(t)}} U_u^{(t)}(b_{u,s}^{(t)}, \mathbf{b_{-u}^{(t)}}, \mathbf{p^{(t)}}) \tag{11a}$$

$$s.t. \quad 0 \leq b_{u,s}^{(t)} \leq I_u^{(t)} \tag{11b}$$

The concept of Nash Equilibrium is adopted towards determining a stable operation point for the system. At the Nash Equilibrium point, any of the end-users has no incentive to change its amount of data offloading, as no end-user can improve its utility by unilaterally changing its data offloading strategy.

Definition 1. *A data offloading vector* $\mathbf{b_u^{(t)}}^* = [b_{1,s}^{(t)*}, ..., b_{u,s}^{(t)*}, ..., b_{|U|,s}^{(t)*}], s \in S$ *is the Nash Equilibrium point of the game* $G = [U, \{A_u^{(t)}\}, \{U_u^{(t)}\}]$, *if for every end-user* u *it holds true that* $U_u^{(t)}(b_{u,s}^{(t)*}, \mathbf{b_{-u}^{(t)}}^*) \geq U_u^{(t)}(b_{u,s}^{(t)}, \mathbf{b_{-u}^{(t)}}^*), \forall b_{u,s}^{(t)} \in A_u^{(t)}$.

In the following analysis, our goal is to show the existence and uniqueness of a Nash Equilibrium for the data offloading game. The necessary and sufficient conditions are: (i) the strategy space $A_u^{(t)}, \forall u \in U$ should be non-empty, convex and compact subset of an Euclidean space $\mathbb{R}^U$; and (ii) the utility function $U_u^{(t)}(b_{u,s}^{(t)}, \mathbf{b_{-u}^{(t)}}, \mathbf{p^{(t)}})$ is continuous in $\mathbf{b_u^{(t)}}$ and quasi-concave in $b_{u,s}^{(t)}$.

Theorem 1. *The Nash Equilibrium point of the game* $G = [U, \{A_u^{(t)}\}, \{U_u^{(t)}\}]$ *exists and the end-user's best response data offloading strategy is given as follows.*

$$BR_u(\mathbf{b_{-u}^{(t)}}^*) = b_{u,s}^{(t)*} = \frac{B_{-u}^{(t)}}{\beta_u}[\frac{\alpha_u \beta_u}{d_u^{(t)} p_s^{(t)}} - 1] \tag{12}$$

where $0 \leq b_{u,s}^{(t)*} \leq I_u^{(t)}$.

Proof of Theorem 1. The strategy space $A_u^{(t)} = [0, I_u^{(t)}]$ represents the amount of data that the end-user u can offload to a MEC server s, thus by defintion it is non-empty, convex, and compact subset of the Euclidean space $\mathbb{R}^U$. In addition, based on Equation (3), the utility function $U_u^{(t)}(b_{u,s}^{(t)}, \mathbf{b}_{-\mathbf{u}}^{(\mathbf{t})}, \mathbf{p}^{(\mathbf{t})})$ is continuous in $\mathbf{b}_{\mathbf{u}}^{(\mathbf{t})}$. Furthermore, we determine the second-order derivative of the utility function $U_u^{(t)}(b_{u,s}^{(t)}, \mathbf{b}_{-\mathbf{u}}^{(\mathbf{t})})$ with respect to $b_{u,s}^{(t)}$, as follows.

$$\frac{\partial^2 U_u^{(t)}(b_{u,s}^{(t)})}{\partial b_{u,s}^{(t)2}} = -\frac{\alpha_u \beta_u^2}{B_{-u}^{(t)2}} \cdot \frac{1}{[\beta_u + \frac{\beta_u b_{u,s}^{(t)}}{B_{-u}^{(t)}}]^2} < 0$$

Given that $\frac{\partial^2 U_u^{(t)}(b_{u,s}^{(t)})}{\partial b_{u,s}^{(t)2}} < 0$, the $U_u^{(t)}(b_{u,s}^{(t)}, \mathbf{b}_{-\mathbf{u}}^{(\mathbf{t})}, \mathbf{p}^{(\mathbf{t})})$ is concave in $b_{u,s}^{(t)}$, it is also quasi-concave in $b_{u,s}^{(t)}$. Therefore, the Nash Equilibrium point of the game $G = [U, \{A_u^{(t)}\}, \{U_u^{(t)}\}]$ exists.

Towards determining the best response strategy of each end-user, we calculate the critical points of the $U_u^{(t)}(b_{u,s}^{(t)}, \mathbf{b}_{-\mathbf{u}}^{(\mathbf{t})}, \mathbf{p}^{(\mathbf{t})})$, as follows.

$$\frac{\partial U_u^{(t)}}{\partial b_{u,s}^{(t)}} = 0 \Leftrightarrow b_{u,s}^{(t)} = \frac{B_{-u}^{(t)}}{\beta_u}\left(\frac{\alpha_u \beta_u}{d_u^{(t)} p_s^{(t)}} - 1\right)$$

The data offloading of each end-user u should satisfy the physical limitations, i.e., $0 \leq b_{u,s}^{(t)} \leq I_u^{(t)}$, thus we have the following cases.

Case A: If $d_u^{(t)} p_s^{(t)} > \alpha_u \beta_u$ then the best response strategy is $b_{u,s}^{(t)*} < 0$. However, since the physical limitation imposed states that $0 \leq b_{u,s}^{(t)}$ and our function is concave, then the best response should be $b_{u,s}^{(t)*} = 0$.

Case B: If $d_u^{(t)} p_s^{(t)} < \frac{\alpha_u \beta_u}{I_u^{(t)} \frac{\beta_u}{B_{-u}^{(t)}} + 1}$ then the best response strategy is $b_{u,s}^{(t)*} > I_u^{(t)}$. However, since the physical limitation imposed states that $b_{u,s}^{(t)} \leq I_u^{(t)}$ and our function is concave, then the best response should be $b_{u,s}^{(t)*} = I_u^{(t)}$.

Case C: If $\frac{\alpha_u \beta_u}{I_u^{(t)} \frac{\beta_u}{B_{-u}^{(t)}} + 1} \leq d_u^{(t)} p_s^{(t)} \leq \alpha_u \beta_u$ then the best response strategy is $0 \leq b_{u,s}^{(t)*} \leq I_u^{(t)}$, which satisfies the physical limitation. In this case, the best response is given by the equation $b_{u,s}^{(t)*} = \frac{B_{-u}^{(t)}}{\beta_u}\left(\frac{\alpha_u \beta_u}{d_u^{(t)} p_s^{(t)}} - 1\right)$. □

Theorem 1 proves the existence of the Nash Equilibrium point of the game G and determines the best response strategy for each end-user $u, u \in U$. In the following theorem, the uniqueness of the Nash Equilibrium point of the game G is examined.

Theorem 2. *The Nash Equilibrium point* $b_{u,s}^{(t)*}, \forall u \in U, s \in S$ *of the game* G *is unique.*

Proof of Theorem 2. Towards proving the uniqueness of the Nash Equilibrium point $b_{u,s}^{(t)*} = BR_u(\mathbf{b}_{-\mathbf{u}}^{(\mathbf{t})*})$, for Cases A and B, the Nash Equilibrium point is trivially unique, while for Case C we should show that the best response strategy $BR_u(\mathbf{b}_{-\mathbf{u}}^{(\mathbf{t})*})$ is a standard function [23]. The properties of a standard function are the following:

- Positivity $\mathbf{f}(\mathbf{x}) > \mathbf{0}$;
- Monotonicity: if $\mathbf{x} \geq \mathbf{x}'$, then $\mathbf{f}(\mathbf{x}) \geq \mathbf{f}(\mathbf{x}')$; and
- Scalability: for all $a > 1$, $a \cdot \mathbf{f}(\mathbf{x}) \geq \mathbf{f}(\mathbf{a} \cdot \mathbf{x})$.

If a fixed point exists in a standard function, then it is unique [23]. Using Equation (12), the above properties of the standard function can be easily shown for the end-user's best response function $BR_u(\mathbf{b}_{-\mathbf{u}}^{(\mathbf{t})*})$. Thus, the Nash Equilibrium point of the game G is unique. □

In conclusion, it is noted that the optimal data offloading of each end-user is given by Equation (12).

4.3. Optimal Pricing of the MEC Servers Computing Services

In this subsection, the optimal pricing of the MEC server's computing services is determined towards maximizing the MEC servers' profit given the offloaded data of the end-users. Combining Equations (7), (10b) and (12), the corresponding optimal pricing problem of the MEC servers can be written as follows.

$$\mathbf{p^{(t)}}^{*} = argmax_{\mathbf{p^{(t)}}} P_s^{(t)}(\mathbf{b^{(t)}}, \mathbf{p^{(t)}}) = (1 - f_s^{(t)}) p_s^{(t)} \sum_{u \in U} [\frac{B_{-u}^{(t)}}{\beta_u} [\frac{\alpha_u \beta_u}{d_u^{(t)} p_s^{(t)}} - 1]] - c_s^{(t)} \sum_{u \in U} [\frac{B_{-u}^{(t)}}{\beta_u} [\frac{\alpha_u \beta_u}{d_u^{(t)} p_s^{(t)}} - 1]] \quad (13)$$

Based on Equation (13), it is observed that the optimal pricing problem of the MEC servers' computing services is a function only of their prices $p_s^{(t)}$.

Theorem 3. *The optimal pricing announced by each MEC server for its computing services given the end-users offloaded data and towards maximizing its own profit is given as follows:*

$$p_s^{(t)*} = [\frac{\alpha_u \beta_u c_s^{(t)} \sum_{u \in U} \frac{B_{-u}^{(t)}}{d_u^{(t)}}}{(1 - f_s^{(t)}) \sum_{u \in U} B_{-u}^{(t)}}]^{1/2} \quad (14)$$

Proof of Theorem 3. Towards determining the optimal pricing announced by each MEC server, we take the first-order derivative with respect to $p_s^{(t)}$ and determine the critical points.

$$\frac{\partial P_s^{(t)}(\mathbf{b^{(t)}}, \mathbf{p^{(t)}})}{\partial p_s^{(t)}} = -\frac{1}{\beta_u}(1 - f_s^{(t)}) \sum_{u \in U} B_{-u}^{(t)} + \frac{c_s^{(t)} \alpha_u}{p_s^{(t)2}} \sum_{u \in U} \frac{B_{-u}^{(t)}}{d_u^{(t)}} = 0$$

Thus, the critical points are given by the following equation.

$$p_s^{(t)*} = [\frac{\alpha_u \beta_u c_s^{(t)} \sum_{u \in U} \frac{B_{-u}^{(t)}}{d_u^{(t)}}}{(1 - f_s^{(t)}) \sum_{u \in U} B_{-u}^{(t)}}]^{1/2}$$

By checking the second-order derivative of $P_s^{(t)}(\mathbf{b^{(t)}}, \mathbf{p^{(t)}})$ with respect to $p_s^{(t)}$, we have:

$$\frac{\partial^2 P_s^{(t)}(\mathbf{b^{(t)}}, \mathbf{p^{(t)}})}{\partial p_s^{(t)2}} = -2c_s^{(t)} \frac{\alpha_u}{p_s^{(t)3}} \sum_{u \in U} \frac{B_{-u}^{(t)}}{d_u^{(t)}} < 0$$

Thus, $p_s^{(t)*}$ maximizes the MEC server's profit $P_s^{(t)}(\mathbf{b^{(t)}}, \mathbf{p^{(t)}})$. □

5. Data Offloading and MEC Server Selection (DO-MECS) Algorithm

In this section, an iterative and low-complexity algorithm is introduced towards realizing the Data Offloading and MEC Server Selection (DO-MECS algorithm). The DO-MECS algorithm consists of two main components. At the first component, the MEC server selection by the end-users is executed following the theory of the stochastic learning automata, as presented in Section 3. Then, at the second component of the DO-MECS algorithm, the end-users' optimal data offloading and the MEC servers' optimal pricing is determined, as presented in Section 4. It is noted that the first part of the DO-MECS algorithm runs at the beginning of each time slot, while the second part of the algorithm runs for multiple iterations within each time slot.

DO-MECS Algorithm

Step 1 (Initialization): At the first time slot $t = 0$, set the initial MEC server selection probability vector as $\mathbf{Pr_u(t = 0)}$, where $Pr_{u,s}(t = 0) = \frac{1}{S}, \forall u \in U, s \in S$.

Step 2 (MEC Server Selection): At the beginning of each time slot ($t > 0$), each end-user chooses a MEC server to offload its data based on its action probability vector $\mathbf{Pr_u(t)}$. If $Pr_{u,s}(t) \geq 0.999$ for all the MEC servers $s, s \in S$, then stop. Otherwise, set $i = 0$, where i denotes the iteration of the second part of the algorithm.

Step 3 (Optimal Data Offloading): Each end-user has been associated with a MEC server and all the MEC servers announce their prices. Each end-user determines its optimal data offloading based on Equation (12).
Step 4 (Optimal Pricing): Given the end-users' offloading data, each MEC server determines the optimal pricing of its computing services based on Equation (14).
Step 5 (Convergence): If $|b_{u,s}^{(t)*}|_{i+1} - b_{u,s}^{(t)*}|_i| \leq \epsilon_1$ and $|p_s^{(t)*}|_{i+1} - p_s^{(t)*}|_i| \leq \epsilon_2, \forall s \in S, u \in U$, where ϵ_1, ϵ_2 (small positive constants) are the convergence control parameters, then stop. Otherwise, go to Step 3.
Step 6 (Update): Update the end-users' action probabilities based on Equations (9a) and (9b) and return to Step 2.

6. Results

In this section, we provide some numerical results illustrating the operation, features and benefits of the proposed DO-MECS framework. In Section 6.1, we focus on the pure operational characteristics of our framework, while in Section 6.2 a comparative evaluation of our approach against alternative methodologies is provided. The algorithm and simulations were implemented in Python (with NumPy), and executed on an Intel Core i5-4300U laptop with CPU@1.90 GHz $\times$ 4 and 8 Gb RAM. Unless otherwise explicitly indicated, a detailed Monte Carlo analysis was executed for all presented numerical results considering averages over 1000 executions.

6.1. Operation of the DO-MECS Framework

Towards illustrating the successful operation of the DO-MECS framework, we performed detailed simulations considering two main cases regarding the end-users that reside within the MEC environment: (a) homogeneous end-users; and (b) heterogeneous end-users, with reference to their sensitivity on the pricing imposed by the MEC servers (i.e., end-user dynamics $d_u^{(t)}$ in Equation (2)). In our simulations, we considered $S = 5$ MEC servers and $U = 100$ end-users, while for demonstration purposes the weights w_1, w_2, w_3 in Equation (4) were considered of same importance, and each one equal to $1/3$. We considered a business perspective with respect to the MEC servers, in the sense that they present different characteristics with respect to parameters such as cost, discount factor, etc. The parameters that characterize the different MEC servers are presented in Table 1. Regarding the communication part of the network operation, each MEC server was assumed to receive data from the users via its own subcarrier. Thus, each user sensed the interference only from the users that were offloading to the same MEC server. It is noted however that, in this study, the transmission power control problem was not treated, and it was assumed that users transmit with fixed power.

Table 1. MEC servers' characteristics.

Server	Cost c	Discount f_s
server 1	0.12	0.05
server 2	0.14	0.04
server 3	0.20	0.02
server 4	0.17	0.03
server 5	0.13	0.05

6.1.1. Homogeneous End-Users

Initially, with respect to the scenario of homogeneous end-users, Figure 3 presents, in a comprehensive manner, indicative numerical results regarding the pure operation of the DO-MECS algorithm, in order to gain some insight about the key operational characteristics and contributions of the various components of our framework. We considered a simplistic demonstration scenario, which however did not harm the validity of the observations. To the contrary, it was selected such that we verified the operational characteristics of our proposed approach, where each user's maximum

amount of data was the same $I_u^{(t)} = 1000$ Bytes. We did not consider or differentiate them based on the nature of the executed tasks or on parameters related to the computing or data intensity. It is also stressed that the focus of this paper and of the corresponding evaluation results is on the decision making process of the data offloading (i.e., server selection and part of data to be offloaded), and not on the actual offloading and/or computation processing itself.

Specifically, Figure 3a presents the relative pricing of each MEC server, i.e., $\frac{\frac{\Sigma_{k \neq s}[(1-f_k^{(t)})]p_k^{(t)}}{K}}{(1-f_s^{(t)})p_s^{(t)}}$, as it was determined at the end of each time slot with respect to the time slots that the DO-MECS algorithm needs to converge. It was observed that in all cases convergence was obtained in fewer than 3000 time slots, while for practical purposes fewer than 2000 time slots were sufficient, corresponding to actual running time of less than 14 s for learning rate $b = 0.2$. Note that significantly lower convergence times could be achieved if higher learning rates were considered, as demonstrated below in Section 6.2.1. Note that the times measured and reported here refer to the convergence of the overall DO-MECS algorithm in our simulation (i.e., decision making process), where the users conclude to a stable selection of MEC servers in order to offload their data to be further processed.

As presented in Figure 3a,b, the greater was the relative pricing for each MEC server, the more attractive it became for the end-users. Server 1 clearly accumulated the majority of the end-users since in Table 1 we notice that Server 1 had both the smallest cost and offered the highest discount compared to the other MEC servers. The same trend and reasoning followed for the rest of the servers. Please note here that, due to the homogeneity of the considered population, each end-user offloaded the same amount of data (in this experiment offloaded its total data, i.e., $I_u^{(t)} = 1000$ Bytes), to the corresponding selected MEC server, as determined by the MEC Server Selection process (Step 2 of DO-MECS Algorithm) based on the theory of the stochastic learning automata (Section 3). In Section 6.1.2, a different scenario with heterogeneous end-users was considered and demonstrated, where the end-users decided to offload different amounts of data, based on the overall system dynamics.

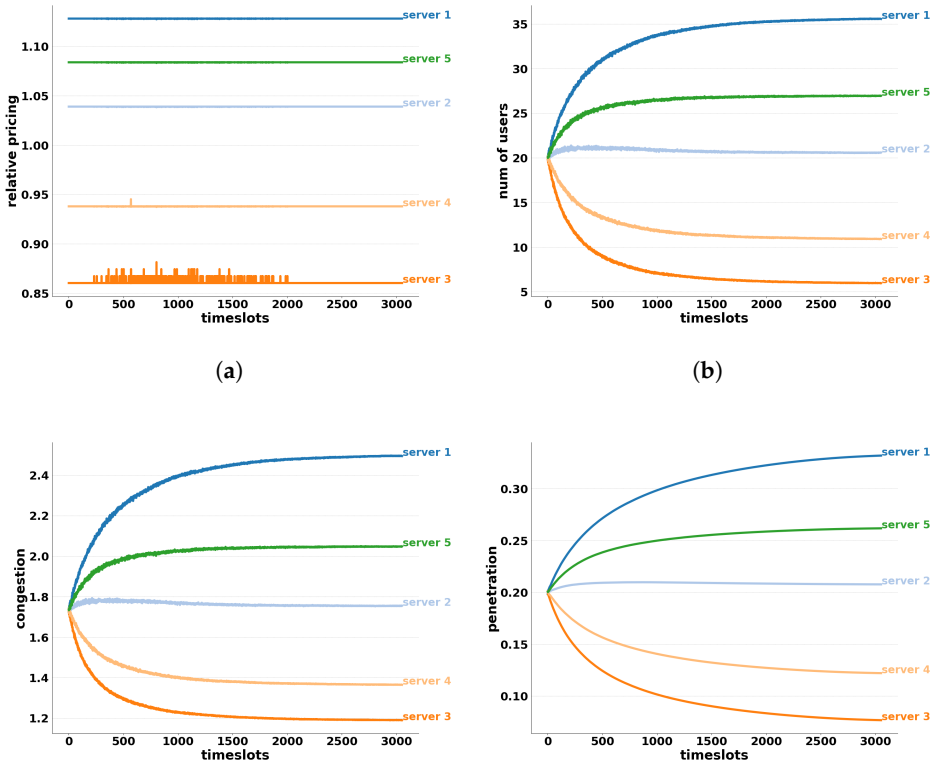
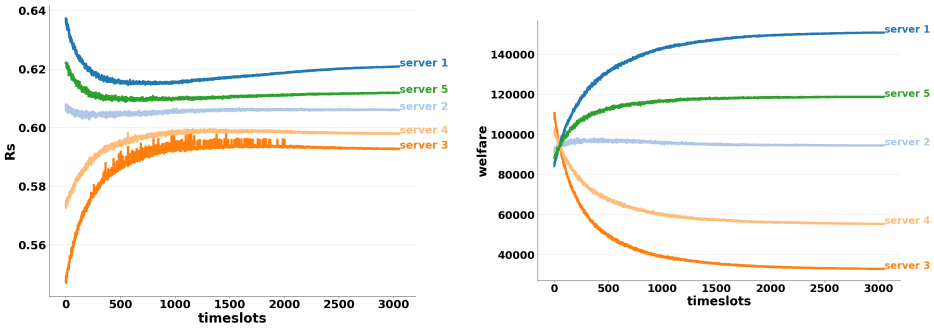


Figure 3. *Cont.*



(**e**) (**f**)

Figure 3. Pure operation of the proposed framework considering homogeneous end-users: (**a**) relative pricing of MEC servers vs. time slots; (**b**) number of end-users per MEC server vs. time slots; (**c**) MEC servers' congestion vs. time slot; (**d**) MEC servers' penetration vs. time slots; (**e**) MEC server's reputation score vs. time slots; and (**f**) MEC server's profit vs. time slots.

As expected, the congestion on each MEC server, i.e., $(1 + CONG_s)^3$, followed the same trend as the number of end-users selecting each MEC server (Figure 3b). The latter observation was expected, as the more end-users selected to offload their data to a MEC server, the more congested that MEC server became (Figure 3c) and a greater penetration, i.e., $\frac{\sum_{t\in\{1,\ldots,T\}}\sum_{u\in U} b_{u,s}^{(t)}}{\sum_{s\in S}\sum_{t\in\{1,\ldots,T\}}\sum_{u\in U} b_{u,s}^{(t)}}$, was achieved by that server. In particular, the MEC servers' penetration in serving the end-users computing demands is presented in Figure 3d. Furthermore, from Equation (4), we observe that the reputation score R_s depends on the relative pricing, the congestion and the penetration of the MEC servers. The R_s essentially controls the probability based on which each end-user will select a server to offload its data. In Figure 3e, the results illustrate that the proposed DO-MECS framework tried to boost "weaker" servers to allow them to gain some traction on the market. Additionally, Figure 3f presents the profit $P_s^{(t)}(\mathbf{b^{(t)}}, \mathbf{p^{(t)}})$ that each MEC server received based on its price announcement and the end-users' data offloading. The results reveal that Server 1 achieved the highest profit due to the combined effect of having the lowest cost (Table 1) and attracting a large number of end-users, despite the fact that it presented the lowest price, as shown in Figure 3a. The same trend was followed from the rest of the servers, which indicates that the announced price by the MEC server was not the only dominant factor in shaping the server's profit, but also the number of end-users that selected to be served by a server was a key parameter in determining the server's overall profit.

6.1.2. Heterogeneous End-Users

We considered the scenario of heterogeneous end-users, i.e., the end-users demonstrate different spending dynamics (i.e., $d_u^{(t)}$) and therefore potentially may offload different parts of their total data $I_u^{(t)}$ to the selected MEC server. Specifically, in Figure 4a, we present the convergence of the amount of offloaded data for 10 indicative end-users from the overall available set in the simulated scenario. The results indicate that, as the end-users had different spending dynamics, the announced price by each MEC server had different impact on each end-user in terms of determining its amount of offloaded data. Due to the differentiation of the end-users' spending dynamics, the MEC servers were motivated to adjust their announced prices to better adapt to the volume of the end-users' offloaded data. The aforementioned behavior is captured in Figure 4b, where it is observed that the "weaker" servers were willing to drop their price to increase their stability and penetration on the market, while the stronger ones increased their price to avoid congestion. Moreover, in Figure 4c,d, the total number of end-users per MEC server and the corresponding amount of offloaded data per MEC server are presented, respectively.

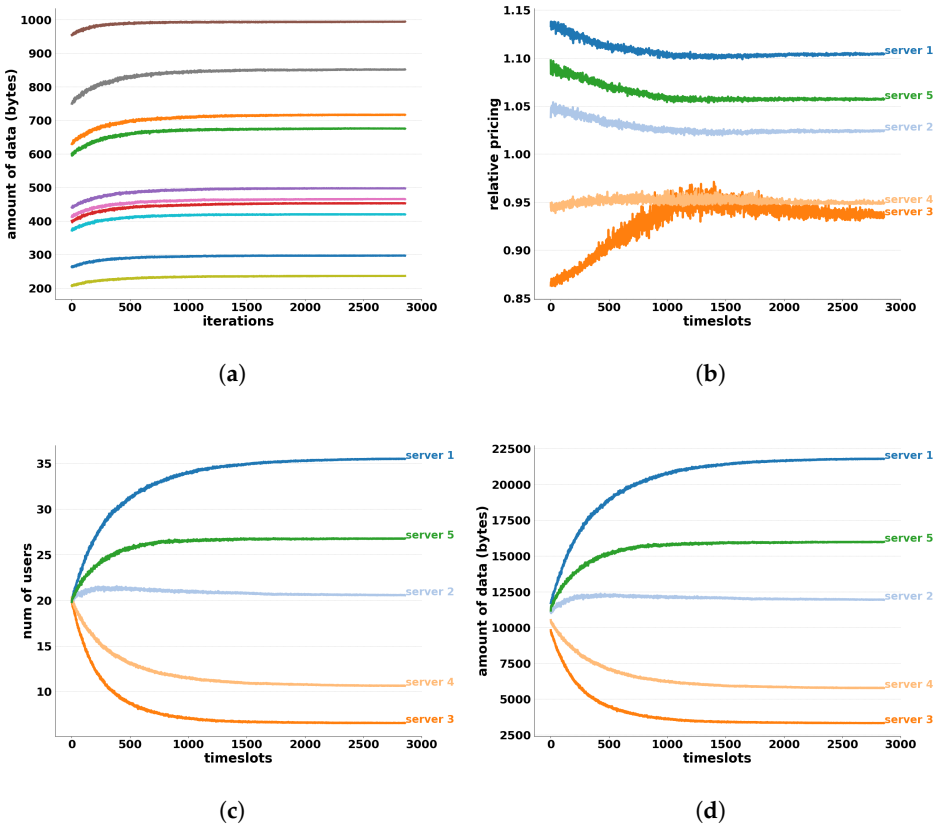


Figure 4. Pure operation of the proposed framework considering heterogeneous end-users: (**a**) end-users' amount of offloaded data vs. time slots; (**b**) relative pricing of MEC servers vs. time slots; (**c**) number of end-users per MEC server vs. time slots; and (**d**) offloaded data to each MEC server vs. time slots.

6.2. Comparative Evaluation

In this section, we present some comparative results of the performance of our proposed framework against some alternative strategies to reveal its benefits and advantages. Initially, in Section 6.2.1, we present the impact of the learning rate parameter of the stochastic learning automata (see Section 3) in the operation of the DO-MECS framework, while, in Section 6.2.2, we evaluate the benefits and drawbacks of different data offloading mechanisms.

6.2.1. Different Learning Rates

As we can see from Equations (6) and (7), the learning rate parameter b is an important factor regarding the convergence of the DO-MECS framework to the optimal stable state. Greater values of the learning rate would lead to faster convergence, however smaller ones allow the end-users to better exploit the available options and ultimately conclude to better states. To demonstrate the above tradeoff, a comparative evaluation between different values of the learning rate was performed. Table 2 shows the average execution time of our DO-MECS framework until convergence was achieved, while Figures 5 and 6 present the average MEC server's profit and the average end-user's utility for different learning rates, respectively. Indeed, it was observed that small values of the learning rate parameter b concluded to slow convergence of the DO-MECS algorithm, however, they allowed the MEC servers and the end-users to achieve higher average profit and higher average utility, respectively. Based on Figures 5 and 6, we can see that the the difference on the convergence state (i.e., average MEC servers' profit and average end-users' utility) between learning rates $b = 0.1$ and $b = 0.2$ was negligible, while the difference in the convergence time was significant. This was also evident from

the execution times presented in Table 2, where for $b = 0.2$ the DO-MECS algorithm converged five times faster than in the case where $b = 0.1$, while, by using a higher value for b (i.e., $b = 0.5$), we could achieve convergence times lower by an order of magnitude. Thus, a learning rate of $b = 0.2$ presented a good balance between optimality and efficiency. The convergence time of the DO-MECS algorithm could be further improved by adopting one of the following strategies and/or a combination of them: (a) Increase the learning rate b. (b) Initiate the algorithm from an "educated" point of MEC servers' selection by the users, i.e., instead each user randomly selecting a MEC server at the first step of DO-MECS algorithm, they can use previous knowledge that would be available in a realistic environment after the initial interaction of the users with the MEC servers. (c) Utilize a more powerful machine with better computational characteristics.

Table 2. Execution time for different learning rate values.

Learning Rate	Execution Time (s)	Number of Timeslots
$b = 0.1$	147.2 s	11053
$b = 0.2$	27.5 s	2959
$b = 0.3$	11.6 s	1357
$b = 0.4$	6.4 s	773
$b = 0.5$	4.2 s	504

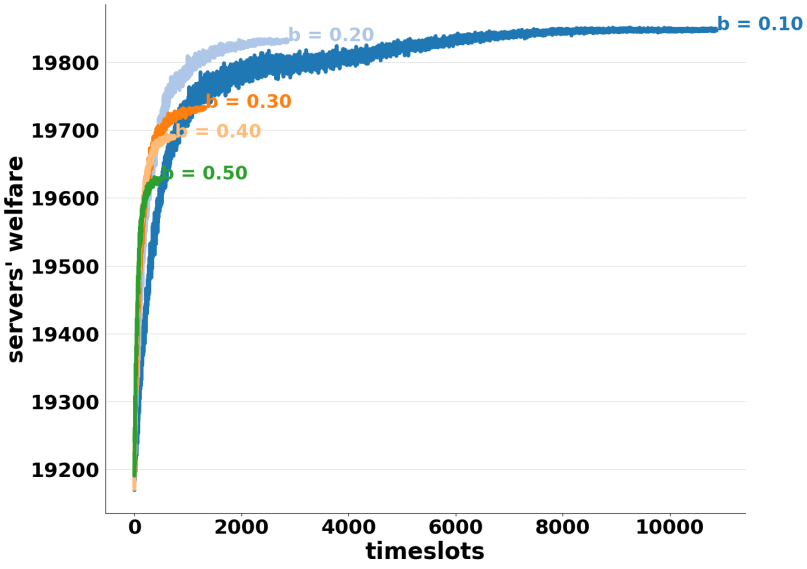


Figure 5. Average MEC servers' profit vs. time slots for different learning rates.

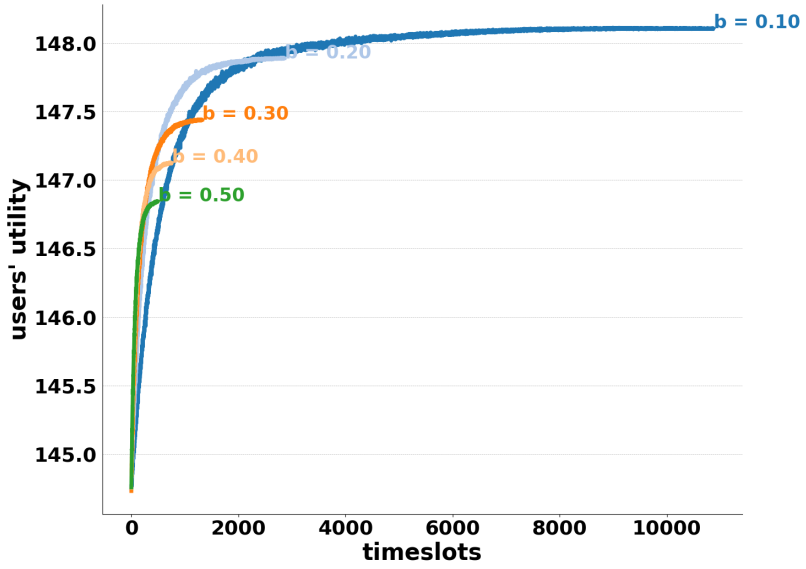


Figure 6. Average end-users' utility vs. time slots for different learning rates.

6.2.2. Different Offloading Mechanisms

Towards evaluating the significance of the game theoretic data offloading mechanism proposed by our DO-MECS framework, a comparison between our mechanism and a computationally simplistic mechanism where each end-user offloads a fixed portion (i.e., percentage) of its data, was performed, while for fairness purposes the rest of our proposed framework (i.e., server selection and optimal pricing mechanisms) was kept intact in all strategies. Specifically, with respect to the alternative data offloading mechanism, three different variations were examined, where the end-users send 25%, 58.6% and 100% of their total data $I_u^{(t)}$, respectively, to the selected MEC servers. It should be noted here that the alternative with fixed portion (i.e., percentage) of 58.6% data offloading of user's maximum amount of data was selected because it corresponds to the same average end-user data offloading, as the one produced by our proposed framework in the considered experiment.

The corresponding comparative results are depicted in Figures 7 and 8, where the average MEC servers' profit and the average end-users' utility, respectively, as a function of the time for the different offloading mechanisms were obtained. In particular, it was evident that as expected the more data the end-users offloaded to the MEC servers, the higher profit the MEC servers experienced. However, this happens at the cost of very low average utility experienced by the end-users, as clearly demonstrated from the curves corresponding to the 100% offloading alternative. Moreover, it was observed that by allowing the end-users to send a constant amount of data without enabling them to dynamically adapt their offloading amount of data based on the system's conditions (as our framework evangelizes), always resulted to significantly lower average end-users' utility. As a result, the proposed DO-MECS framework offered incentives to the end-users to participate in the non-cooperative data offloading game in order to dynamically and autonomously determine the optimal amount of data, while the MEC servers experienced the best levels of profit that they could achieve based on the decisions of their customers, i.e., end-users.

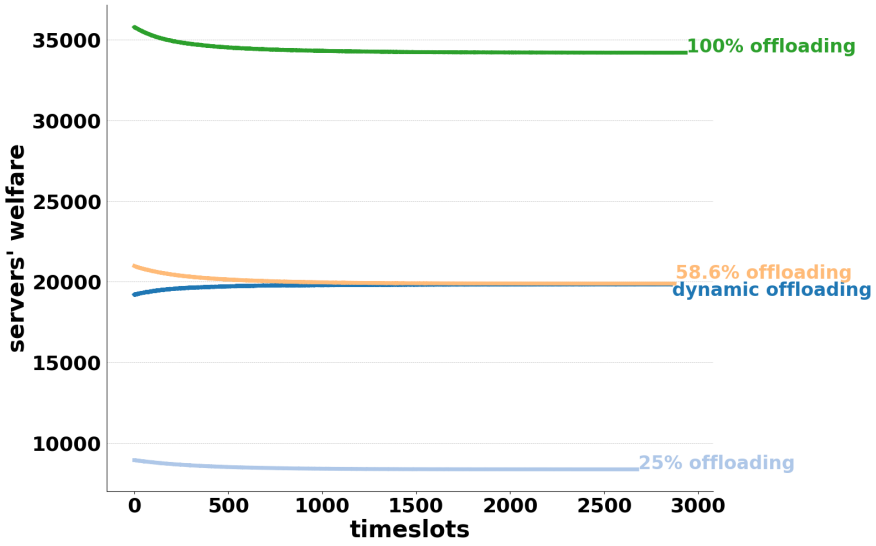


Figure 7. Average MEC servers' profit vs. time slots for different offloading mechanisms.

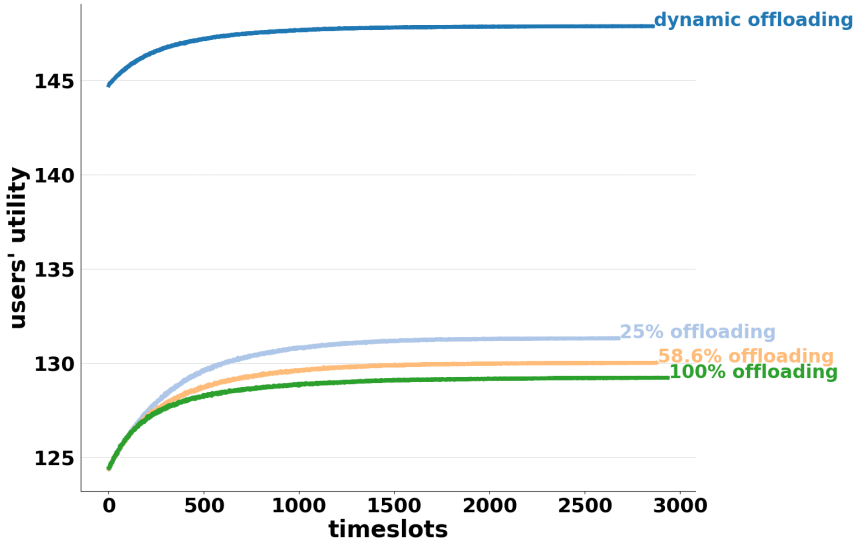


Figure 8. Average end-users' utility vs. time slots for different offloading mechanisms.

7. Conclusions

In this paper, the joint problem of MEC server selection by the end-users, along with their optimal data offloading and the optimal price setting by the MEC servers is studied in a multiple MEC servers and multiple end-users environment. The flexibility and programmability offered by the SDN technology, enables the realistic implementation of the proposed framework. In particular, the MEC server selection part of the framework is based on a reinforcement learning technique adopting the theory of the stochastic learning automata. The end-users optimal data offloading and the MEC servers' optimal pricing of their computing services is formulated as a two-layer optimization problem. At the first layer, a non-cooperative game among the end-users of each server is formulated towards maximizing the perceived satisfaction of each end-user, as expressed by an appropriately formulated utility function. The existence and uniqueness of the game's NE point is shown, thus concluding

to the end-users' optimal data offloading strategy. At the second layer of the proposed framework, an optimization problem of each MEC server's profit is formulated and the corresponding optimal price of its computing services is determined. A low-complexity Data Offloading and MEC Server Selection (DO-MECS) algorithm is introduced to realize the overall framework. The operation and performance of the proposed framework was extensively evaluated through modeling and simulation, while the presented detailed numerical results demonstrate its performance and benefits in the examined setting.

Our current and future work contains the testing of the proposed framework in a realistic testbed environment, while the proposed framework will be extended to include additional socio-physical parameters in the MEC server's reputation score, e.g., trust level of the MEC server, security and privacy preserving characteristics, and others.

Author Contributions: All authors contributed extensively to the work presented in this paper. G.M. contributed to the design of the algorithm, developed the code of the overall framework, executed the evaluation experiment and contributed to the discussions and analysis of the comparative evaluation results. P.A.A. had the original idea on which we have based our current work and contributed to the design of the proposed framework. E.E.T. and S.P. were responsible for the overall orchestration of the performance evaluation work and had the overall coordination in the writing of the article.

Funding: The research of Eirini Eleni Tsiropoulou was conducted as part of the UNM Research Allocation Committee award and the UNM Women in STEM Faculty Development Fund.

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

References

1. Barbarossa S.; Sardellitti, S.; Di Lorenzo, P. Joint allocation of computation and communication resources in multiuser mobile cloud computing. In Proceedings of the IEEE 14th Workshop on Signal Processing Advances in Wireless Communications (SPAWC), Darmstadt, Germany, 16–19 July 2013; pp. 26–30.
2. Mao, Y.; Zhang, J.; Song, S.; Letaief, K.B. Stochastic joint radio and computational resource management for multi-user mobile-edge computing systems. *IEEE Trans. Wirel. Commun.* **2017**, *16*, 5994–6009. [CrossRef]
3. Munoz, O.; Pascual-Iserte, A.; Vidal, J. Optimization of radio and computational resources for energy efficiency in latency-constrained application offloading. *IEEE Trans. Veh. Technol* **2015**, *64*, 4738–4755. [CrossRef]
4. You, C.; Huang, K.; Chae, H.; Kim, B.-H. Energy-efficient resource allocation for mobile-edge computation offloading. *IEEE Trans. Wirel. Commun.* **2017**, *16*, 1397–1411. [CrossRef]
5. Yu, Y.; Zhang, J.; Letaief, K.B. Joint subcarrier and cpu time allocation for mobile edge computing. In Proceedings of the IEEE Global Communications Conference (GLOBECOM), Washington, DC, USA, 4–8 December 2016; pp. 1–6.
6. Wang, Y.; Sheng, M.; Wang, X.; Wang, L.; Li, J. Mobile-edge computing: Partial computation offloading using dynamic voltage scaling. *IEEE Trans. Commun.* **2016**, *64*, 4268–4282. [CrossRef]
7. Guo, S.; Xiao, B.; Yang, Y.; Yang, Y. Energy-efficient dynamic offloading and resource scheduling in mobile cloud computing. In Proceedings of the 35th Annual IEEE International Conference on Computer Communications, San Francisco, CA, USA, 10–14 April 2016, pp. 1–9.
8. Chen, M.-H.; Liang, B.; Dong, M. Joint offloading and resource allocation for computation and communication in mobile cloud with computing access point. In Proceedings of the IEEE Conference on Computer Communications, Atlanta, GA, USA, 1–4 May 2017; pp. 1–9.
9. Chen, X.; Jiao, L.; Li, W.; Fu, X. Efficient multi-user computation offloading for mobile-edge cloud computing. *IEEE/ACM Trans. Netw.* **2016**, *24*, 2795–2808. [CrossRef]
10. Chen, X. Decentralized computation offloading game for mobile cloud computing. *IEEE Trans. Parallel Distrib. Syst.* **2015**, *26*, 974–983. [CrossRef]
11. Jošilo, S.; Dán, G. A game theoretic analysis of selfish mobile computation offloading. In Proceedings of the IEEE Conference on Computer Communications, Atlanta, GA, USA, 1–4 May 2017; pp. 1–9.
12. Apostolopoulos, P.A.; Tsiropoulou, E.E.; Papavassiliou, S. Game-Theoretic Learning-Based QoS Satisfaction in Autonomous Mobile Edge Computing. In Proceedings of the IEEE Global Information Infrastructure and Networking Symposium (GIIS), Thessaloniki, Greece, 23–25 October 2018; pp. 1–5.

13. Zhang, K.; Mao, Y.; Leng, S.; Maharjan, S.; Zhang, Y. Optimal delay constrained offloading for vehicular edge computing networks. In Proceedings of the IEEE International Conference on Communications (ICC), Paris, France, 21–25 May 2017; pp. 1–6.
14. Xie, K.; Wang, X.; Xie, G.; Xie, D.; Cao, J.; Ji, Y.; Wen, J. Distributed multi-dimensional pricing for efficient application offloading in mobile cloud computing. *IEEE Trans. Serv. Comput.* **2016**. [CrossRef]
15. Jararweh, Y.; Ahmad Doulat, A.D.; Mohammad Alsmirat, M.A.A.; Benkhelifa, E. SDMEC: Software defined system for mobile edge computing. In Proceedings of the 2016 IEEE International Conference on Cloud Engineering Workshop (IC2EW), Berlin, Germany, 4–8 April 2016; pp. 88–93.
16. Wang, J.; Li, D. Adaptive Computing Optimization in Software-Defined Network-Based Industrial Internet of Things with Fog Computing. *Sensors* **2018**, *18*, 2509. [CrossRef] [PubMed]
17. Porambage, P.; Okwuibe, J.; Liyanage, M.; Ylianttila, M.; Taleb, T. Survey on multi-access edge computing for internet of things realization. *IEEE Commun. Surv. Tutor.* **2018**, *20*, 2961–2991. [CrossRef]
18. Baktir, A.C.; Ozgovde, A.; Ersoy, C. How can edge computing benefit from software-defined networking: A survey, use cases, and future directions. *IEEE Commun. Surv. Tutor.* **2017**, *19*, 2359–2391. [CrossRef]
19. Ali, S.; Ghazal, M. Real-time Heart Attack Mobile Detection Service (RHAMDS): An IoT use case for Software Defined Networks. In Proceedings of the 30th IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), Windsor, ON, Canada, 30 April–3 May 2017; pp. 1–6.
20. Huang, A.; Nikaein, N.; Stenbock, T.; Ksentini, A.; Bonnet, C. Low Latency MEC Framework for SDN-based LTE/LTE-A Networks. In Proceedings of the IEEE International Conference on Communications (ICC), Paris, France, 21–25 May 2017; pp. 1–6.
21. Hossain, M.S.; Xu, C.; Li, Y.; Pathan, A.-S.K.; Bilbao, J.; Zeng, W.; El Saddik, A. Impact of Next-Generation Mobile Technologies on IoT-Cloud Convergence. *Commun. Mag.* **2017**, *55*, 18–19. [CrossRef]
22. Tsiropoulou, E.E.; Kousis, G.; Thanou, A.; Lykourentzou, I.; Papavassiliou, S. Quality of Experience in Cyber-Physical Social Systems Based on Reinforcement Learning and Game Theory. *Future Internet* **2018**, *10*, 108. [CrossRef]
23. Tsiropoulou, E.E.; Vamvakas, P.; Papavassiliou, S. Joint customized price and power control for energy-efficient multi-service wireless networks via S-modular theory. *IEEE Trans. Green Commun. Netw.* **2017**, *1*, 17–28. [CrossRef]