Future Generation Computer Systems 100 (2019) 531–541

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Resource allocation and computation offloading with data security for mobile edge computing

Ibrahim A. Elgendy [a], Weizhe Zhang [a,b,*], Yu-Chu Tian [c], Keqin Li [d]

[a] School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China
[b] Peng Cheng laboratory, Shenzhen, China
[c] School of Electrical Engineering and Computer Science, QUT, Brisbane QLD, Australia
[d] Department of Computer Science, State University of New York, New Paltz, NY 12561, USA

HIGHLIGHTS

- Resource allocation and computation-offloading model for MEC system is proposed.
- An AES is introduced as a security layer to protect the computation tasks.
- An offloading algorithm is designed to find the optimal computation offloading decision.
- Minimization of time and energy consumption of the entire system is the main goal.

ARTICLE INFO

Article history:
Received 10 November 2018
Received in revised form 25 April 2019
Accepted 14 May 2019
Available online 23 May 2019

Keywords:
Computation offloading
Internet of Things (IoT)
Mobile-edge computing
Optimization
Security

ABSTRACT

With the considerable growth of mobile users (MUs) and IoT devices, complex applications and multimedia services are rapidly increasing, thereby requiring additional computations and high data communication. However, these devices are still resource-constrained with limited computation power and energy. Furthermore, security is considered a critical issue for sensitive information communication. This study presents a multiuser resource allocation and computation offloading model with data security to address the limitations of such devices. First, the computation and radio resources are jointly considered for multiuser scenarios to guarantee the efficient utilization of shared resources. In addition, an AES cryptographic technique is introduced as a security layer to protect sensitive information from cyber-attacks. Furthermore, an integrated model, which jointly considers security, computation offloading, and resource allocation, is formulated to minimize time and energy consumption of the entire system. Finally, an offloading algorithm is developed with detailed processes to determine the optimal computation offloading decision for MUs. Simulation results show that our model and algorithm can significantly improve the performance of the entire system compared with local execution and full offloading schemes.

1. Introduction

In recent years, mobile network and wireless technology development has resulted in various powerful mobile applications and multimedia services, such as video games, face recognition, augmented reality, healthcare, and natural language processing [1,2]. In addition, most of these applications and services typically require intensive computation and high processing, which are incompatible with devices due to their limited resources [3–5].

Mobile cloud computing is considered a prominent solution that address the limitations of mobile users (MUs), in which mobile applications' intensive computations will be offloaded to a centralized cloud via a wireless channel to mitigate the load and extend the battery life [6–9]. However, high latency is one of the main shortcomings of centralized cloud computing, in which the MUs may take a long time for lagged data transmission and have difficulty addressing real-time applications [10]. Moreover, security is considered another evident challenge of the cloud paradigm, in which the applications' data and services are vulnerable to different threats during cloud transmission.

To address the challenges of mobile cloud computing, researchers have realized that utilizing the resources and services that are nearest to the MUs is considered a cost-efficient solution with low latency. This finding has led to a new paradigm called mobile edge computing (MEC) [11]. MEC is defined as a network

* Corresponding author at: School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China.
E-mail addresses: ibrahim.elgendy@hit.edu.cn (I.A. Elgendy), wzzhang@hit.edu.cn (W. Zhang), y.tian@qut.edu.au (Y.-C. Tian), lik@newpaltz.edu (K. Li).

https://doi.org/10.1016/j.future.2019.05.037
0167-739X/

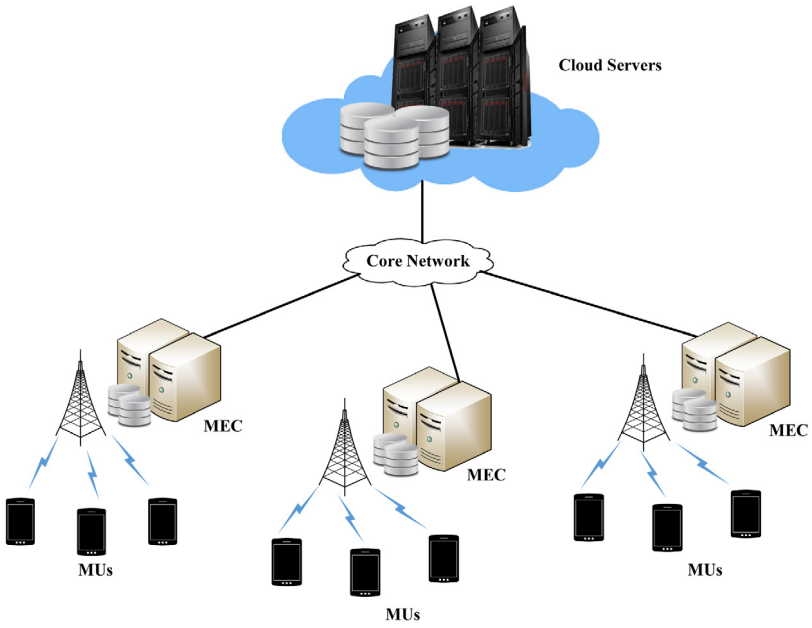

Fig. 1. Mobile edge computing architecture.

architecture that pushes cloud computing capabilities at edge nodes that are close to MUs and connected to cloud servers via a core network, as shown in Fig. 1. In addition, MEC is characterized by high bandwidth and low latency, which allows the execution of real-time applications [12,13].

Many studies have applied the concept of computation offloading on the mobile edge computing paradigm to minimize energy consumption, satisfy delay requirements, allocate radio resources efficiently, maximize total revenue, maximize system utility, and/or the reduce total cost of MUs. In addition, studies that specifically address the security for fog computing and MEC are fewer than those that focus on mobile cloud computing. Furthermore, although most MUs and multimedia services generate a large amount of data and information that can be transmitted over mobile cellular networks, the security issue for multiusers in MEC is not recognized well [14,15]. This concern motivates our study to address the security issue of multiuser computation offloading and allocate radio and computation resources efficiently.

Therefore, in this study, we propose to consider resource allocation and computation offloading jointly to minimize the overall energy and time consumption of the entire system. In addition, an efficient and secured layer is introduced to protect the computation tasks and their related data from any cyber-attack before edge server transmission using an AES cryptographic technique. The main contributions of this study are summarized as follows:

- An AES cryptographic technique is introduced as a security layer to protect the computation tasks and their related data from any cyber-attack before data are transferred to the MEC server.
- An optimization problem, which jointly considers computation offloading and resource allocation, is formulated to minimize time and energy consumption of the entire system. This optimization problem is a 0–1 linear optimization problem, which is considered NP-hard. Therefore, the solution can be calculated via the branch and bound method.
- An offloading algorithm is designed to determine the optimal computation offloading decision for all MUs in the MEC system.
- Simulation results are presented to show the effectiveness of our proposed model and algorithm on the performance of the entire system in terms of energy and time.

The remainder of this paper is organized as follows. Section 2 reviews related works. Section 3 describes our system model in terms of resource allocation, computation offloading, and security, and the optimization problem is formulated. The algorithm design is presented in Section 4. In Section 5, simulation experiments are conducted to demonstrate our offloading model and algorithm. Finally, Section 6 concludes the paper.

2. Related work

In recent years, the MEC system has used numerous strategies and objectives for offloading computation tasks from MUs to the edge server to address the challenges of MUs [12,13,16]. In this section, we present some of the common computation offloading models based on objective attributes.

(1) Minimize energy

An energy-efficient computation offloading approach is designed in [17], in which computation task offloading and its data transmission are jointly considered in the offloading decision. In addition, the minimization of energy consumption of the offloading system under delay constraints is the main goal of this approach. The results prove that the proposed approach has enhanced energy efficiency, especially with large number of MUs. Nevertheless, the proposed method of this work offloads all the application for remote execution, which is considered resource consuming, such that a large amount of data will be transferred over the network. In addition, the application data must be safe while transmitting; thus, an efficient security technique should be applied.

The cooperative computing concept is investigated in [18–20] to minimize the energy consumption of MUs. Cao et al. [18] consider user cooperation in the MEC system, in which communication and computation resource allocation are jointly optimized. Particularly, the authors consider a basic three-node MEC system in utilizing joint communication and computation cooperation to minimize the total energy consumption. Furthermore, in [19] and [20], the cooperation is presented between MUs and wireless

sensor nodes, respectively, in which an opportunistic helper is used as a cooperative computing. In [19], each MU can utilize noncausal information on the helper-CPU state to utilize the random computation resources efficiently by offloading the computation of input data. In [20], the authors assume that the mobile applications are partitioned into independent tasks that will be offloaded and executed efficiently and jointly consider computation and communication resources. However, the main drawback in [17–19] is identifying the proper computing devices in proximity (helper node) while guaranteeing that the processed data will be delivered back to the source mobile device. Therefore, quality of service and quality of experience (QoE) for users can be hardly guaranteed. In addition, these studies only consider the single MU scenario.

Recently, an innovative framework with a wireless-powered multiuser MEC system is developed in [21]. This framework allows flexibility for each user to offload all or part of the computation tasks using a time-division multiple access protocol. In addition, an optimal resource allocation approach is proposed to improve the performance of the entire system in terms of energy consumption. Nonetheless, the near-far problem is considered one of the main drawbacks of this work, in which the farther MU can harvest less energy and require to communicate in long distances, thereby consuming additional energy. In addition, each MU cannot obtain the result until all the other devices offload their computation task because TDMA is used. Consequently, handling real-time applications is difficult when the number of users increase.

(2) Minimize latency

Kao et al. [22] propose a polynomial-time approximation approach for the task dependency of general mobile applications to minimize the latency of a mobile application while addressing resource utilization constraints. In addition, an online learning algorithm is proposed with a provable performance guarantee. The results show that compared with the heuristic method, the proposed approach not only improves the latency performance but also scale well with the problem size. However, the researchers do not consider the optimization of data traffic incurred during task execution and the application data are not protected from attacks during transfer to the server. In addition, this approach only works on one offloading request scenario.

The minimization of mobile applications' latency is also the main goal in [23,24] and [25]. In [23], a distributed cloud-aware power control algorithm, which is considered a suitable algorithm for real-time applications, is proposed. From the MU perspective, the work in [24] proposes an autonomic computation offloading framework based on deep reinforcement Q-learning approach for handling resource requirement and mobility issues in the MEC system. Furthermore, Liu et al. [25] develop a one-dimensional search algorithm based on Markov chain theory. In addition, an optimal computation task-scheduling policy and an optimization problem are formulated for the MEC system. Nevertheless, the application data are not protected from cyber-attacks during transfer to the server, which is considered the main disadvantage of [23–25]. In addition, the work in [23] is insufficient for large MU mobility. Furthermore, in [24], the cost increased by increasing the number of nodes and applying machine learning.

A conceptual fog-computing framework is proposed in [26], in which an arbitrary number of fog devices are combined to provide a suitable resource provisioning solution at nearby places and offload the computation tasks among these devices. In addition, the service placement is formulated as an optimization problem to minimize the latency of the communication and maximize the resource utilization of fog devices. Furthermore, genetic and first fit algorithms are applied to solve this problem. However, identifying the appropriate IoT devices in proximity and the communication between them while guaranteeing that the processed data will be delivered back to the source mobile device is considered the main drawback of this method.

Ren et al. [27] propose three different models for multiuser MEC based on the TDMA technique. The first two models compress the application's data locally on the MU and the edge server, whereas the application's data are partially compressed at the MU and at the edge server in the third model. The results show that the last model can improve the performance of mobile applications in terms of end-to-end latency compared with the other two models. However, advanced wireless communication techniques, which can improve edge computing performance are not considered this work.

(3) Minimize energy and latency

An energy-latency task and resource allocation algorithm is proposed in [28], in which the computation offloading policy and the network traffic are jointly optimized for the mobile cloud system using the Lyapunov optimization technique. Furthermore, the interference problem among the MUs is mitigated. Finally, the simulation results show that compared with other existing approaches, the proposed algorithm not only reduces the energy consumption but also satisfies the delay requirements for different types of application task. Nevertheless, this approach is an unrealistic system model because only one MU is considered in the system. In addition, the offloaded data are vulnerable to cyber-attacks.

Wang et al. [29] design a partial computation offloading approach that jointly considers communication and computation resources for single and multiple server scenarios. In addition, energy consumption and application latency are minimized by proposing an energy-optimal partial offloading algorithm and a local optimal algorithm, respectively. The computation offloading and the transmission power are jointly optimized for the MEC system in [30], in which a tradeoff between mobile energy consumption and application latency is achieved. Furthermore, a message-passing framework is proposed to decrease the complexity of computation offloading by leveraging the application's topological structure. However, these studies [29,30] do not protect the transmitted data from cyber-attacks.

The power-delay tradeoff for the multiuser MEC systems is also the main objective of [31] and [32]. Mao et al. [31] formulate a power consumption minimization problem with the application's task buffer stability constraints. In addition, an online algorithm is presented to determine the manner in which the transmission power and bandwidth for the MUs' offloading will be allocated and find the CPU frequencies for local execution. Chen et al. [32] jointly optimize a mobile application's computation offloading decisions and the communication and computation resource allocation via an efficient three-step algorithm. However, the interference caused by the transmission of other users, which greatly affects system performance, is not considered in [31]. Moreover, the processing deadline for user computation tasks is not imposed in [31,32].

(4) Maximize revenue or system utility

The maximization of the total revenue of a network is considered as the main objective of [33], in which resource allocation, computation offloading decision, and content caching strategy are jointly formulated as an optimization problem. Furthermore, an alternating direction algorithm based on the multiplier method is proposed to solve this problem efficiently. However, this work does not consider any user deadlines for performing computation tasks, which can be impractical. In addition, wireless resource allocation is not involved.

Lyu et al. [34] develop a semi-distributed and heuristic computation offloading decision algorithm for multiuser MEC system to maximize system utility in terms of the QoE of MUs. In addition,

the authors jointly optimize computation and communication resources, as well as the computation offloading decision. Moreover, the authors in [35] study the cooperation among geographically distributed service providers to provide the MUs with efficient resources and reduce the computation and communication delay in comparison with traditional approaches. Furthermore, the authors aim to maximize system utility, in which the cost related to resource utilization and VM migration is considered. Nonetheless, only the resource allocation is considered in [34], which is less effective on the system. In addition, the resource cooperation type (i.e., local or remote) is not considered in [35].

(5) Security

Security is considered an important issue that should be addressed for MUs where most multimedia services and applications provide private and sensitive information that can be misused by attackers. Therefore, Zhang et al. [36] present a CloudSafe system, which provides a cloud-based, personal, digital asset-safe service to keep user data distributed among more than one cloud provider using erasure coding and cryptography. In addition, the CloudSafe system uses AES cryptography algorithm to protect the confidentiality of user data. Nevertheless, this approach requires more management effort, in which data are distributed over more than one server. Another work proposes an efficient confidentiality-based cloud storage framework for protecting data storage on cloud computing on the basis of its confidentiality degree [37]. First, the user specifies whether the confidentiality level of data is basic, confidential or highly confidential. Then, the data are protected using HTTPS and TLS if it is a basic level or is encrypted using AES-128 or AES-256 if it is confidential or highly confidential, respectively. However, this framework adds burden on users to classify data.

Furthermore, a verifiable outsourced multiauthority access control approach is proposed in [38], where most encryption and decryption computations are outsourced to fog devices. In addition, an efficient user and attribute revocation method is designed to address the revocation issue and verify the computation results. However, the encryption and decryption processes of the ciphertext-policy, attribute-based encryption system, in which this approach is based, are time-consuming.

Recently, an efficient and secured framework for the computation offloading of mobile IoT devices is proposed in [39,40]. In [39], Almajali et al. use edge nodes as a third party for applying security, in which the framework is divided into three main steps. First, the mobile device is registered with a cloud provider by sending a message that contains user ID, password, and application requirements. Second, the cloud provider communicates with one or more edge nodes to find the nearest one that satisfies the application requirements and initiates the communication between this node and the mobile device. Third, the application data are encrypted and transmitted to the edge node, which will send them to the cloud provider for processing. Meng et al. [40] propose a secure and cost-efficient offloading approach for mobile cloud computing. This framework optimizes the security and performance tradeoff of the system by using a hybrid continuous-time Markov chain and queueing model, respectively. Nevertheless, in [39,40], management and processing are done by the cloud provider which require long communication and cause delay. In addition, users must repeat the second step in [39] due to its mobility or edge node failure, which is considered time-consuming when the mobile device mobility is large.

Table 1 summarized the mentioned related works and shown the main weakness.

The preceding review of related work shows that computation offloading has been investigated with different objectives for MUs. In addition, studies that specifically address the security for fog computing and MEC are fewer than those that focus on mobile cloud computing. Furthermore, the security issue for multiusers in MEC is not addressed; in which most MUs and multimedia services generate substantial data and information that can be transmitted over mobile cellular networks. This limitation has motivated our research to address the security issue for multiuser computation offloading and allocate the radio and computation resources efficiently.

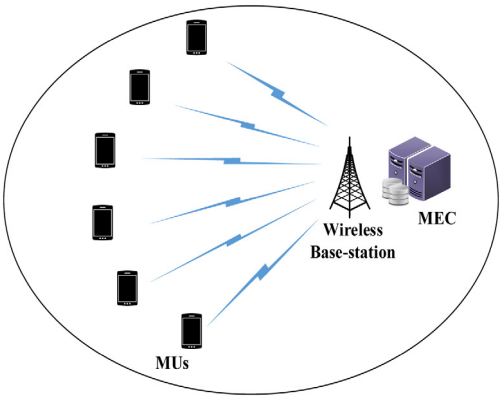

Fig. 2. System model.

3. System model

This section introduces the system model adopted in this study. We consider a multiuser computation offloading for MEC systems, where $\mathcal{M} = \{1, 2, \ldots, N\}$ is defined as a set of MUs that is connected with a single wireless base station, and the MEC server is placed at the wireless base station, as shown in Fig. 2. In addition, each MU has a different computation and processing capacity requirement that needs to be completed. Hence, we use a tuple $\{\beta_i, \delta_i, \Gamma_i\}$ to represent the task requirement for each MU i, where β_i, δ_i, and Γ_i are the number of CPU cycles, the data size, and the completion deadline, respectively. This information can be obtained using program profilers [41].

Similar to many previous works [42,43], our simulation work on a quasistatic[1] scenario where the number of MUs N remains unchanged during the offloading period, while it may change across different periods. Communication, computation, and security play a key role in MEC; thus, communication, computation, and security models are presented in detail, followed by the formulation of the optimization problem of our model.

The notations used in this study are summarized in Table 2.

3.1. Communication model

We initially introduce the communication model in MEC system, in which our environment has N MUs that are associated with a single base station and the edge server via a wireless channel. In addition, each MU has a different computation and processing capacity requirement that must be completed. We denote $\alpha_i \in \{0, 1\}$ as a binary computation offloading decision for each MU i, where $(\alpha_i = 0)$ indicates that user i will decide to compute the computation task locally on its own MU, and $(\alpha_i = 1)$ implies that the MU i will decide to offload the computation task to the edge server via a wireless channel. Thus, we have $\alpha = \{\alpha_1, \alpha_2, \ldots, \alpha_N\}$ as the offloading decision profile for all MUs. Furthermore, we assume that the intracellular interference for the uplink transmission is well-mitigated using orthogonal frequency

[1] The general case where the mobility of mobile users from one base station to another during the offloading period will be considered in the future work.

Table 1
Comparison between existing models.

Ref	Objective	Proposed solution	Offloading nature	No. of MUs.	No. of server nodes	Weakness
[17]	Minimize energy	Energy efficient computation offloading algorithm.	Full	Multi MUs	Single node	• Resource consuming where all the application will be offloaded. • No security applied.
[18]	Minimize energy	Energy efficient computation and communication cooperation approach.	Partial	Single MU	Single node	
[19]	Minimize energy	Energy-efficient co-computing policies using non-causal CPU-state information.	Partial	Single MU	Single node	• QoS and QoE can be hardly guaranteed. • No security applied.
[20]	Minimize energy	Energy-efficient co-computing using wireless sensors.	Partial	Single MU	Multiple nodes	
[21]	Minimize energy	Innovation framework that Integrate the wireless power transfer and MEC for multi-users system.	Partial	Multi MUs	Single node	• Near-far problem. • Inconvenient for real-time applications.
[22]	Minimize latency	Polynomial-time approximation approach for general mobile application's task dependency.	Partial	Multi MUs	Single node	• Only work on the one offloading request scenario. • No security applied.
[23]	Minimize latency	Distributed cloud-aware power control algorithm for real-time applications.	Full	Single MU	Multiples nodes	• Insufficient for large mobility. • No security applied.
[24]	Minimize latency	Autonomic computation offloading framework for handling the resource requirements and mobility issue.	Full	Multi MUs	Multiple nodes	• More costly by increasing the number of nodes and applying machine learning. • No security applied.
[25]	Minimize latency	One-dimensional search algorithm for finding the optimal scheduling policy.	Full	Single MU	Single node	• No security applied.
[26]	Minimize latency	A conceptual fog-computing framework for providing a suitable resource provisioning solution.	Full	Multi MUs	Multiple nodes	• QoS and QoE can be hardly guaranteed.
[27]	Minimize latency	A Partial offloading model for allocating the communication and computation resource.	Partial	Multi MUs	Single node	• Advanced wireless communication techniques are not handled. • No security applied.
[28]	Minimize energy and latency	An energy-latency task and resource allocation algorithm for deciding the offloading policy.	Partial	Single MU	Single node	• Unrealistic system model where only one MU is considered. • No security applied.
[29]	Minimize energy and latency	Partial computation offloading scheme using dynamic voltage scaling	Partial	Single MU	Single & Multiple nodes	• No security applied.
[30]	Minimize energy and latency	Message-passing framework using the structure of the call graphs for decreasing the complexity of the computation offloading.	Partial	Single MU	Single node	• No security applied.
[31]	Minimize energy and latency	A stochastic task arrival model for solving the energy-latency trade-off problem.	Partial	Multi MUs	Single node	• Interference that is caused by the transmission of other users is not considered.
[32]	Minimize energy and latency	A three-step algorithm for optimizing the offloading decisions and the resource allocation.	Partial	Multi MUs	Single node	• Processing deadline for the user's computation tasks is not imposed
[33]	Maximize Revenue	Optimal approach for considering resource allocation, computation offloading and content caching.	Partial	Multi MUs	Single node	• Computation task deadlines are not considered. • Wireless resource allocation was not involved.
[34]	Maximize system utility	Semi-distributed and heuristic computation offloading decision algorithm for optimizing the resource allocation and offloading decisions.	Partial	Multi MUs	Single node	• Only the resource allocation is considered which is less effective.
[35]	Maximize system utility	Game-theoretic algorithm for handling the cooperation between geographically distributed service providers.	Full	Multi MUs	Multiple nodes	• Resource cooperation type is not considered. • No security applied.
[36]	Security	CloudSafe system to protect the confidentiality of user data using AES algorithm.	Full	Single MU	Multiple nodes	• More management efforts where the data are distributed over more than one server.
[37]	Security	Efficient framework for protecting the data storage on the cloud computing.	Full	Single MU	Single node	• Add an extra burden on the user to classify the data.
[38]	Security	Verifiable outsourced multi-authority access control approach is proposed	Full	Single MU	Multiple nodes	• This approach is time-consuming
[39]	Security	Efficient and secured framework for mobile IoT devices.	Full	Single MU	Single node	• Require a long communication and cause delay where the management and processing are done by a cloud provider. • Time-consuming for the large mobility scale where the mobile users must repeat the second step for each mobility change.
[40]	Security	A secure and cost-efficient offloading approach.	Full	Single MU	Single node	• Require a long communication and cause delay where the management and processing are done by a cloud provider.

Table 2
Notations.

Notation	Description
$\mathcal{M}$	The set of MUs
N	Total number of MUs
i	Refer to the ith MU
β_i	CPU cycles to accomplish the tasks
δ_i	Data size for computation tasks
Γ_i	Completion deadline for MU i
α_i	Computation offloading decision of MU i
r_i	Uplink data rate for MU i
B	Total System Bandwidth
p_i	Transmission power of MU i
τ_i	Channel gain of the base station
θ_0	Density of noise power of channel
ζ_i	Energy consumed per CPU cycle
f_i^l	Computational capability of MU i
f_i^s	Computation resource assigned to MU i
F	MEC server computational capability
S_i	Security decision of MU i
C_i	CPU cycles required to encrypt the tasks
D_i	CPU cycles required to decrypt the tasks
t_i^{off}	Time for offloading tasks on the edge server
t_i^{proc}	Time for executing tasks on the edge server
e_i^{off}	Energy for offloading tasks at the edge server
t_i^{secu}	Time for applying the security layer
e_i^{secu}	Energy for applying the security layer
T_i^l	Time for executing tasks on the MU i
E_i^l	Energy for executing tasks on the MU i
T_i^s	Time for executing tasks at the edge server
E_i^s	Energy for executing tasks at the edge server
H_i^l	Overhead for executing tasks on the MU i
H_i^s	Overhead for executing tasks at the edge server
w_i^t	Local computational time weight of MU i
w_i^e	Local computational energy weight of MU i

for different user transmissions in the same cell [44]. Therefore, the uplink data rate for each MU i can be obtained as follows [45]:

$$r_i = B \,.\, log_2(1 + \frac{p_i \tau_i^2}{\theta_0 B}) \tag{1}$$

where B denotes the system bandwidth, p_i denotes the transmission power of MU i, θ_0 denotes the density of noise power of channel and τ_i denotes the channel gain of the base station.

Consequently, when all MUs offload their computation tasks via the wireless access channel simultaneously during a computation offloading period, a constraint is the bandwidth limit, as shown as follows:

$$\sum_{i=1}^{N} \alpha_i r_i \leq B, \qquad \forall i \in N \tag{2}$$

In this study, we assume that the overhead consumption for the output in terms of energy and time is neglected; this assumption is caused by the data size being smaller after task execution than it is before execution, and the downlink rate from the server is higher than the uplink rate [34].

3.2. Computation model

This subsection introduces the computation offloading model. First, as mentioned previously, our simulation has N MUs that have a different computation and processing capacity requirement that must be completed. For each task, we use a tuple $\{\beta_i, \delta_i, \Gamma_i\}$ to represent the task requirement for each MU i, where β_i, δ_i, and Γ_i represent the number of CPU cycles, the data size, and the completion deadline, respectively. The computation overhead in terms of execution time and energy consumption for local and MEC execution approaches will be discussed later in detail.

3.2.1. Time and energy for local computing

For the local computing approach, each MU i executes all the computation tasks locally. In addition, each MU has different computational capabilities. As a result, the total execution time and energy consumption for executing the computation task locally can be respectively expressed as:

$$T_i^l = \frac{\beta_i}{f_i^l} \tag{3}$$

$$E_i^l = \zeta_i \beta_i \tag{4}$$

where f_i^l denotes the computational capability (CPU cycles per seconds) of MU i, and ζ_i is a coefficient, that denotes the consumed energy per CPU cycle. On the basis of the measurement obtained in [46], we set $\zeta_i = 10^{-11}(f_i^l)^2$, where the energy consumption is a superlinear function of mobile device frequency [42].

3.2.2. Time and energy for edge server computing

For the edge computing approach, each MU i will offload its computation task to the edge server via a wireless channel depending on the computation offloading decision α_i. With regard to the communication model and the uplink data rate for each user i, the total execution time for the remote execution of the computation task on the edge server (i.e., task offloading and task processing) can be expressed as:

$$t_i^{off} + t_i^{proc} = \frac{\delta_i}{r_i} + \frac{\beta_i}{f_i^s} \tag{5}$$

where f_i^s denotes the computational capability of the edge server assigned to MU i. However, we assume that the computation resource allocated to each user is proportional to the computation capability of that user.

Furthermore, the energy consumption for the remote execution of the computation task on the edge server (i.e., task offloading only) can be expressed as:

$$e_i^{off} = p_i t_i^{off} \tag{6}$$

3.3. Security model

Each MU i can transmit the computation task's data to the MEC server via a wireless channel in the offloading case. In addition, these data are vulnerable to different types of threat and attack while transmitting to the edge server. As a result, an AES cryptographic technique is introduced as a secured layer to protect the computation tasks and their related data against the different types of threat where it is considered the most popular and standard symmetric cryptography algorithm for encrypting and decrypting data confidently and also is efficient in terms of security and performance [47,48].

For the security model, we denote $S_i \in \{0, 1\}$ as a binary security decision for each MU i, where $(S_i = 0)$ indicates that user i will offload the computation task and their data without encryption, and $(S_i = 1)$ implies that the MU i will encrypt the computation task and their data using the security layer before transmitting to the edge server. Subsequently, upon receiving the data, the edge server will decrypt the data, execute the computation tasks, and send the result back to the MU. Thus, we have $S = \{S_1, S_2, \ldots, S_N\}$ as the security decision profile for all MUs. This decision is manually made by each MU on the basis of the privacy requirements for the application data. Therefore, in view of the security layer, the additional overhead in terms of time and

Table 3
Simulation parameters.

Parameter	Value
Total number of MUs N	30
System bandwidth B	20 MHz
Transmission power of MU p_i	100 mW
Background noise θ_0	−100 dBm
weights w_i^t, w_i^e	{0, 0.2, 0.5, 0.8, 1.0}
Data size for computation tasks δ_i	5000 kB
CPU cycles to accomplish tasks β_i	1000 Megacycles
Computation capability of MU f_i^l	{0.4, 0.5, ..., 1.0} GHz
MEC server capability F	10 GHz

energy for the remote execution of computation tasks on the MEC server can be respectively expressed as:

$$t_i^{secu} = t_i^{enc+dec} = S_i\alpha_i(\frac{C_i}{f_i^l} + \frac{D_i}{f_i^s}) \tag{7}$$

$$e_i^{secu} = e_i^{enc} = S_i\alpha_i(\zeta_i C_i) \tag{8}$$

where C_i and D_i represent the total number of CPU cycles required to encrypt and decrypt the data of the computation task on the MU and at the edge server, respectively [49].

In view of the previous subsections, in which communication, computation, and security models are considered, the total time and energy for the execution of the computation tasks of MU i can be respectively expressed as:

$$T_i^s = t_i^{secu} + t_i^{off} + t_i^{proc} \tag{9}$$

$$E_i^s = e_i^{secu} + e_i^{off} \tag{10}$$

On the basis of Eqs. (3), (4), (9), and (10), the total overhead to execute all tasks locally and remotely on the MU and at the edge server in terms of execution time and energy consumption can be respectively calculated as:

$$H_i^l = w_i^t T_i^l + w_i^e E_i^l \tag{11}$$

$$H_i^s = w_i^t T_i^s + w_i^e E_i^s \tag{12}$$

where w_i^t and $w_i^e \in [0, 1]$ denote the weighting parameters of execution time and energy consumption for MU i's decision making, respectively. For example, $w_i^e = 1$ and $w_i^t = 0$ if the mobile battery of user i is in a low state, whereas $w_i^e = 0$ and $w_i^t = 1$ if the MU i is running a real-time application sensitive to the delay (e.g., video streaming); different values are set to w_i^e and w_i^t for different objectives.

The computation capabilities assigned to all MUs must be quantified to construct the system model. They should be capped by the available resources (i.e., the computation capacity of the server CPU) on the edge server denoted by F, as shown as follows:

$$\sum_{i=1}^{N} \alpha_i f_i^s \le F, \qquad \forall i \in N \tag{13}$$

Furthermore, from the developed system model, the computation offloading, resource allocation, and security will be formulated as an optimization problem. This approach aids us develop resource allocation and computation offloading with a data security model for a multiuser MEC system, which will be discussed in the next subsection.

3.4. Problem formulation

In this subsection, the integration model of resource allocation, computation offloading, and security for a multiuser MEC system is formulated as an optimization problem. Here, we aim to minimize the energy consumption and execution time of the entire system is considered. Thus, the problem is formulated as follows:

$$\begin{aligned}
\min_{\alpha} \quad & \left[\sum_{i=1}^{N} \alpha_i H_i^s + \sum_{i=1}^{N} (1-\alpha_i) H_i^l\right] \\
\text{s.t} \quad & \left[\alpha_i E_i^s + (1-\alpha_i) E_i^l\right] \le E_i^l, && \forall i \in N \quad C1 \\
& \sum_{i=1}^{N} \alpha_i r_i \le B, && \forall i \in N \quad C2 \\
& \sum_{i=1}^{N} \alpha_i f_i^s \le F, && \forall i \in N \quad C3 \\
& T_i \le \Gamma_i, && \forall i \in N \quad C4 \\
& \alpha_i \in \{0, 1\}, && \forall i \in N \quad C5
\end{aligned} \tag{14}$$

The objective function computes the minimal consumption in terms of energy and time for the entire system. Constraint C1 ensures that the total energy consumption for the remote execution for all the MUs does not exceed the total energy consumption for local execution. Constraint C2 controls the channel bandwidth capacity. Constraint C3 characterizes the upper limit of the CPU capacity of the edge server. Constraint C4 determines the deadline requirement for the task completion. Finally, Constraint C5 guarantees that the computation offloading decision variable is binary.

Our problem in Eq. (14) is an integer linear optimization problem, in which the objective function and all the constraints are linear. Thus, the optimal value of α^* variable can be obtained using the branch and bound method.

4. Algorithm design

An algorithm is designed to obtain an optimal solution to the constrained optimization problem in Eq. (14), as shown in Algorithm 1, which is self-explanatory. The algorithm provides detailed processes to derive an optimal computation offloading decision for multiple users.

The time complexity of this algorithm is represented by $O(N)$, where N denotes the total number of MUs. Therefore, the algorithm does not consume additional MU resources.

In the designed algorithm, all MUs initialize their computation offloading decision as $\alpha_i(0) = 0$, which implies local execution. Then, each MU sends its tasks requirements $\{\beta_i, \delta_i, \Gamma_i, C_i, D_i, p_i\}$ and its computational capability f_i^l to the nearest MEC server. Subsequently, the MEC server obtains the number of users and then calculates the data rate for each MU on the basis of Eq. (1). Moreover, the MEC server determines the optimal computation offloading decision α_i values for each MU i, which is achieved by solving the optimization problem in Eq. (14). Finally, each MU receives a decision from the server, thereby minimizing the overall time and energy consumption of the entire system.

5. Simulation results and discussion

This section demonstrates our proposed computation offloading model through simulations. The performance of the model is evaluated for the following four different scheduling policies.

1. **Local Execution**: No offloading exists. All tasks are executed locally on MUs ($\alpha_i = 0, \forall i \in N$).
2. **Full Offloading**: All MUs offload their tasks to the MEC server for remote execution ($\alpha_i = 1, \forall i \in N$).

Algorithm 1 Multi-Users Computation Offloading Decision

1: **Initialization**: Each MU i initializes the offloading decision with $\alpha_i = 0, \forall i \in N$
2: **for all** each MU i and at given time slot t **do**
3: **Transmit** the computation task requirements $\{\beta_i, \delta_i, \Gamma_i, C_i, D_i, p_i\}$ as well as the computational capability of each MU f_i^l to the edge server.
4: **Get** the number of users and its computation's requirements and then calculate the uplink data rate r_i for each MU using Eq. (1).
5: **Find** the optimal computation offloading decision values α_i for each MU which minimizes the overall consumption in terms of energy and time of the entire system using Eq. (14).
6: **Send** the optimal computation offloading decision values α_i to each MU.
7: **end for**

3. **Unsecured Model**: The offloading decision is made on the basis of the current environment without adding any security layer for the protection of application data ($\alpha_i \in \{0, 1\}, \forall i \in N$).
4. **Secured Model**: The offloading decision is made on the basis of the current environment with the addition of a security layer for the protection of the application data ($\alpha_i \in \{0, 1\}, \forall i \in N$).

Our simulations are run on a MATLAB-based simulator using a computer equipped with Intel® Core(TM) i7-4770 CPU with 3.4 GHz frequency and 16 GB RAM capacity and runs Windows 10 Professional 64-bit platform. We consider an MEC system with 30 MUs. The CPU computational capability of each MU is randomly assigned from the set $\{0.4, 0.5, \ldots, 1.0\}$ GHz to account for the heterogeneous computing capability of MUs. The CPU computational capability of the MEC server is set to be 10 GHz. Accordingly, the computing speed of the MEC server is also set to be 10 GHz. As an example of a sophisticated application, we consider the face recognition application in [32], in which the data size for the computation offloading is 5000 kB. The total number of CPU cycles required to complete this task is 1000 megacycles. The decision weight of the execution time w_i^t for each MU is assigned randomly from the set $\{0, 0.2, 0.5, 0.8, 1.0\}$. The decision weight for energy consumption is calculated from $w_i^e = 1 - w_i^t$. For example, if $w_i^t = 1$ ($w_i^e = 0$), then a user is only concerned about the execution time (energy consumption); if $w_i^t = 0.5$, then MU i concerned about execution time and energy consumption. Moreover, the number of CPU cycles required to encrypt and decrypt the transmitted data is assumed to be 100 megacycles. Furthermore, the security decision is set randomly in our simulation. Other settings on communication and computation are listed in Table 3. We perform our simulations for 50 runs, from which averaged results can be derived.

In our simulations, the offloading percentages of the MUs for secured and unsecured models versus the total number of MUs are shown in Fig. 3. The offloading percentage of the MUs remains 100% when the number of MUs is less than 15 and begins to decline as the number of MUs increases from 15. When the number of MUs becomes sufficiently large, more devices will offload their computation tasks to the MEC server, thereby causing severe interference among one another. In this case, our model automatically customizes the offloading requests generated by MUs to minimize the overall energy consumption and execution time. As a result, the offloading percentage decreases (Fig. 3).

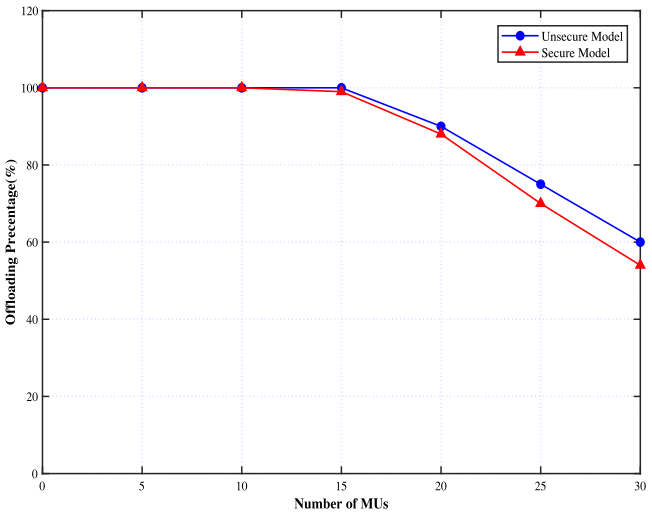

Fig. 3. Offloading percentage of MUs.

The average execution time of offloading tasks to an MEC server versus the number of MUs for the four different scenarios is shown in Fig. 4. The figure shows a summation of two times, that is, the communication time for transferring the task data through the wireless channel, and the computation time for running the tasks on the MEC server and encrypting and decrypting in the secured model case. As shown in Fig. 4, when the number of MUs is less than 15, full offloading with or without security is better than local execution. However, as the number of MUs further increases, the performance of full offloading declines compared with local execution, whereas our offloading model remains better than local execution. This result is due to the fact the time for communication over the shared communication channels increases with the number of MUs.

Fig. 5 depicts the average energy consumption of offloading tasks to an MEC server versus the number of MUs. From this figure, energy consumption increases linearly with the number of MUs. When the number of users is greater than 15, our proposed model can reduce the overall energy consumption. This phenomenon is explained as follows. In the full offloading scenario, all users compete for limited communication and computation resources. As a result, the energy consumption of offloading tasks will increase drastically when the number of MUs becomes large.

Fig. 6 shows the total overhead versus the number of MUs. From the figure, with approximately 15 MUs, the total overhead of full offloading becomes more than that of local execution. In comparison, our model with or without security addition can maintain a lower overhead than local execution. This result is obtained because our proposed model selects some computation tasks to offload while rejecting others in an optimal manner, thereby minimizing the overall overhead in terms of time and energy of the entire system.

Finally, our model with and without security addition is compared with two related works (i.e., [17] and [21]) which are more closer to our work. Fig. 7(a) depicts the energy consumption versus the number of MUs for our model and the two related works. The energy consumption for our model is less than other two works. Furthermore, in comparison with the local execution, Fig. 7(b) shows that the average energy saving of our model with and without security addition and of [17] and [21] is 17.19%, 14.65%, 14.55%, and 14.62%, respectively. This result is obtained because our model offloads only the intensive computation tasks instead of all the application, such as in [17], which is considered

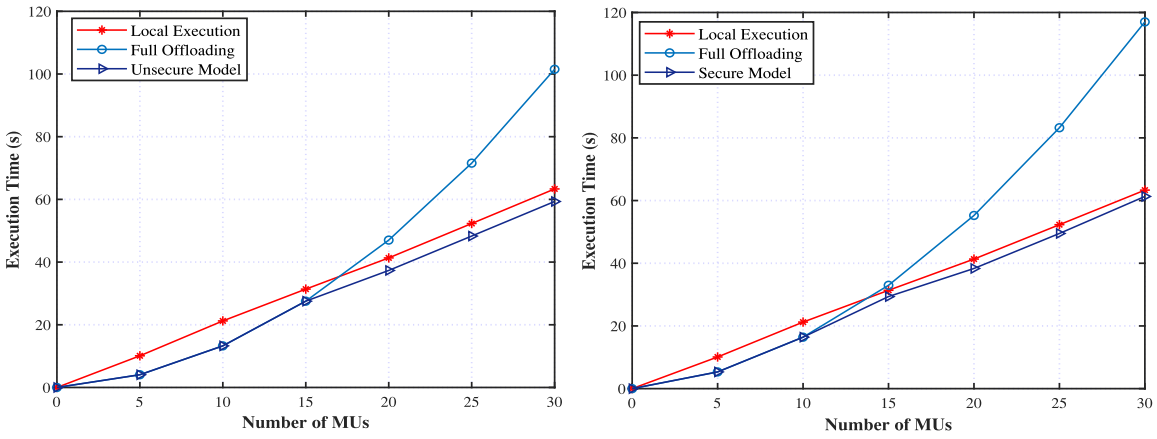

Fig. 4. Execution time versus number of MUs.

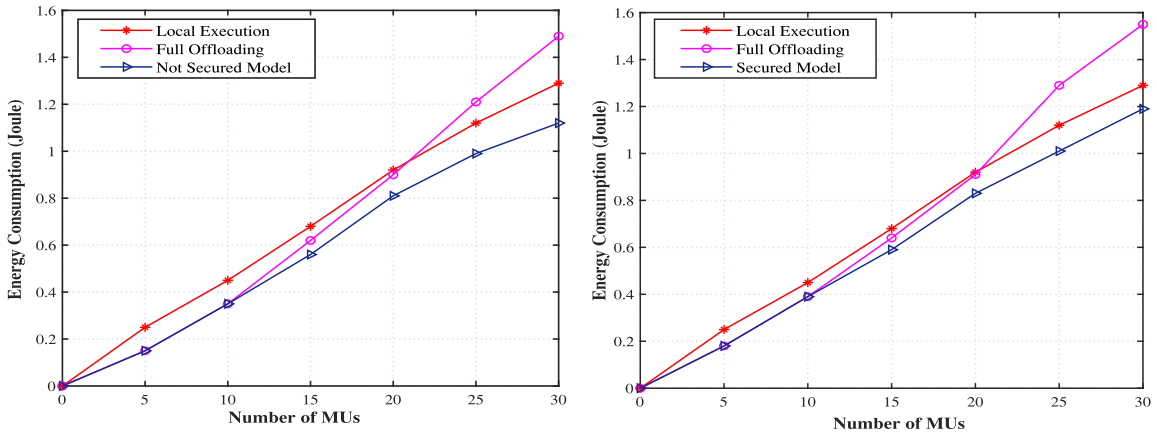

Fig. 5. Energy consumption versus number of MUs.

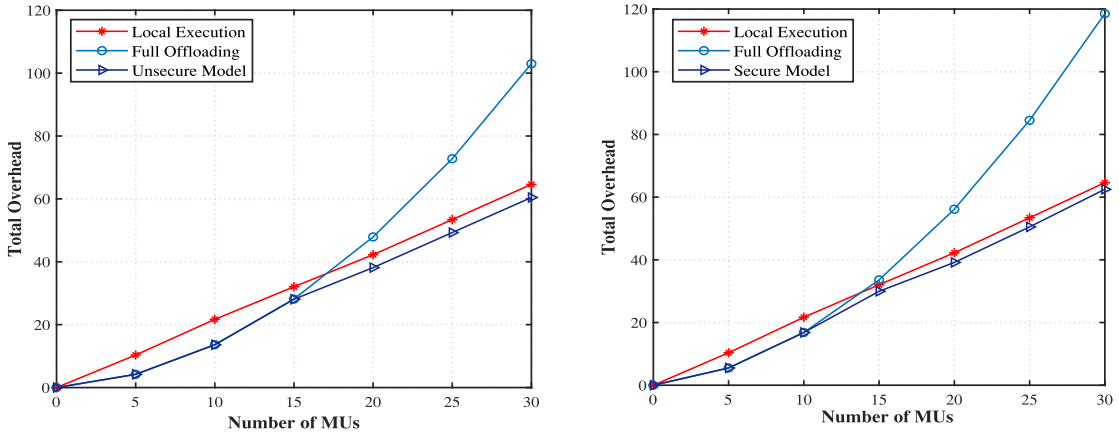

Fig. 6. Total overhead versus number of MUs.

resource consuming. In addition, our model uses the orthogonal frequency for different user transmissions in the same cell, thereby mitigating the intracellular interference and allowing the MUs to offload simultaneously. Moreover, our model selects some MUs to offloaded their computation tasks while rejecting others in an optimal manner.

6. Conclusions

In this study, we propose an efficient resource allocation and computation offloading model for the multiuser MEC system. In addition, an AES cryptographic technique is introduced as a security layer to protect sensitive information from cyber-attacks. Furthermore, an optimization problem, which can jointly consider resource allocation, computation offloading decision, and data security issues, is formulated to minimize the overall consumption in terms of energy and task execution delay for MUs. To address this problem efficiently, an optimal computation offloading algorithm is then developed with detailed processes to determine the optimal offloading decision for the MUs. Furthermore, simulation experiments of the proposed model are presented in comparison with the local execution and the full

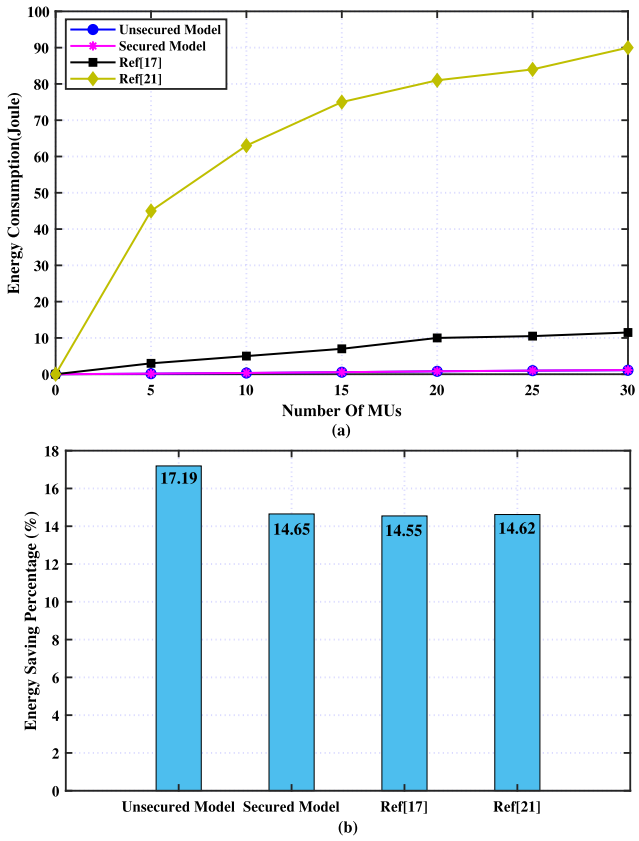

Fig. 7. Comparison with two of related works.

offloading model with or without security addition. Finally, the experimental results demonstrate that our proposed model with or without security layer addition can minimize the total overhead in terms of time and energy with approximately 15 MUs. These results are obtained because our proposed model selects some computation tasks to offload while rejecting others in an optimal manner, thereby minimizing the overall overhead of the entire system.

For future work, we will use an effective compression layer to reduce the size of the offloading computation task's data in the low bandwidth state, thereby improving the performance of the entire system. Another direction is to consider a more general case where MUs may depart and leave dynamically within a computation offloading period, which will be interesting and technically challenging.

Acknowledgments

This work is supported by the National Key Research and Development Plan, China under grant (No. 2016YFB0800801), the National Natural Science Foundation of China (NSFC) under grant (No. 61672186 and 61872110), and the Australian Research Council (ARC) under grant (No. DP170103305). Professor Zhang is the corresponding author.

Conflict of interest

None.

Declaration of competing interest

The authors declared that they had no conflicts of interest with respect to their authorship or the publication of this article.

References

[1] J. Gubbi, R. Buyya, S. Marusic, M. Palaniswami, Internet of things (iot): A vision, architectural elements, and future directions, Future Gener. Comput. Syst. 29 (7) (2013) 1645–1660.
[2] G. Fortino, R. Gravina, A. Guerrieri, G.D. Fatta, Engineering large-scale body area networks applications, in: International Conference on Body Area Networks, 2013, pp. 363–369.
[3] N. Vallina-Rodriguez, J. Crowcroft, Energy management techniques in modern mobile handsets, Ieee Commun. Surv. Tutor. 15 (1) (2013) 179–198.
[4] R. Yadav, W. Zhang, O. Kaiwartya, P.R. Singh, I.A. Elgendy, Y. Tian, Adaptive energy-aware algorithms for minimizing energy consumption and SLA violation in cloud computing, IEEE Access 6 (2018) 55923–55936.
[5] C. Savaglio, G. Fortino, M. Zhou, Towards interoperable, cognitive and autonomic iot systems: An agent-based approach, in: Internet of Things, 2017, pp. 58–63.
[6] N. Fernando, S.W. Loke, W. Rahayu, Mobile cloud computing: A survey, Future Gener. Comput. Syst. 29 (1) (2013) 84–106.
[7] I.A. Elgendy, M. El-kawkagy, A. Keshk, An efficient framework to improve the performance of mobile applications, Int. J. Digit. Content Technol. Appl. (JDCTA) 9 (5) (2015) 43–54.
[8] A.A. Ateya, A. Muthanna, A. Vybornova, P. Darya, A. Koucheryavy, Energy-aware offloading algorithm for multi-level cloud based 5g system, in: Internet of Things, Smart Spaces, and Next Generation Networks and Systems, Springer, 2018, pp. 355–370.
[9] M.A. Elgendy, A. Shawish, M.I. Moussa, MCACC: New approach for augmenting the computing capabilities of mobile devices with cloud computing, in: 2014 Science and Information Conference, IEEE, 2014, pp. 79–86.
[10] Y. Jararweh, A. Doulat, O. AlQudah, E. Ahmed, M. Al-Ayyoub, E. Benkhelifa, The future of mobile cloud computing: integrating cloudlets and mobile edge computing, in: International Conference on Telecommunications, IEEE, 2016, pp. 1–5.
[11] Y. Mao, C. You, J. Zhang, K. Huang, K.B. Letaief, A survey on mobile edge computing: The communication perspective, IEEE Commun. Surv. Tutor. 19 (4) (2017) 2322–2358.
[12] N. Abbas, Y. Zhang, A. Taherkordi, T. Skeie, Mobile edge computing: A survey, IEEE Internet Things J. 5 (1) (2018) 450–465.
[13] P. Mach, Z. Becvar, Mobile edge computing: A survey on architecture and computation offloading, IEEE Commun. Surv. Tutor. 19 (3) (2017) 1628–1656.
[14] A.N. Khan, M.M. Kiah, S.U. Khan, S.A. Madani, Towards secure mobile cloud computing: A survey, Future Gener. Comput. Syst. 29 (5) (2013) 1278–1299.
[15] R. Roman, J. Lopez, M. Mambo, Mobile edge computing, fog et al.: A survey and analysis of security threats and challenges, Future Gener. Comput. Syst. 78 (2018) 680–698.
[16] J. Wang, J. Pan, F. Esposito, P. Calyam, Z. Yang, P. Mohapatra, Edge Cloud Offloading Algorithms: Issues, Methods, and Perspectives, arXiv preprint arXiv:1806.06191, 2018.
[17] K. Zhang, Y. Mao, S. Leng, Q. Zhao, L. Li, X. Peng, L. Pan, S. Maharjan, Y. Zhang, Energy-efficient offloading for mobile edge computing in 5g heterogeneous networks, IEEE Access 4 (2016) 5896–5907.
[18] X. Cao, F. Wang, J. Xu, R. Zhang, S. Cui, Joint Computation and Communication Cooperation for Mobile Edge Computing, arXiv preprint arXiv:1704.06777, 2018.
[19] C. You, K. Huang, Exploiting non-Causal CPU-state information for energy-efficient mobile cooperative computing, IEEE Trans. Wirel. Commun. 17 (6) (2018) 4104–4117.
[20] Z. Sheng, C. Mahapatra, V.C.M. Leung, M. Chen, P.K. Sahu, Energy efficient cooperative computing in mobile wireless sensor networks, IEEE Trans. Cloud Comput. 6 (1) (2018) 114–126.
[21] F. Wang, J. Xu, X. Wang, S. Cui, Joint offloading and computing optimization in wireless powered mobile-edge computing systems, IEEE Trans. Wirel. Commun. 17 (3) (2018) 1784–1797.
[22] Y. Kao, B. Krishnamachari, M. Ra, F. Bai, Hermes: Latency optimal task assignment for resource-constrained mobile computing, IEEE Trans. Mob. Comput. 16 (11) (2017) 3056–3069.
[23] P. Mach, Z. Becvar, Cloud-aware power control for real-time application offloading in mobile edge computing, Trans. Emerg. Telecommun. Technol. 27 (5) (2016) 648–661.
[24] M.G.R. Alam, M.M. Hassan, M.Z. Uddin, A. Almogren, G. Fortino, Autonomic computation offloading in mobile edge for iot applications, Future Gener. Comput. Syst. 90 (2019) 149–157.
[25] J. Liu, Y. Mao, J. Zhang, K.B. Letaief, Delay-optimal computation task scheduling for mobile-edge computing systems, in: 2016 IEEE International Symposium on Information Theory (ISIT), 2016, pp. 1451–1455.
[26] O. Skarlat, M. Nardelli, S. Schulte, M. Borkowski, P. Leitner, Optimized iot service placement in the fog, Service Oriented Comput. Appl. 11 (4) (2017) 427–443.

[27] J. Ren, G. Yu, Y. Cai, Y. He, Latency optimization for resource allocation in mobile-edge computation offloading, IEEE Trans. Wirel. Commun. 17 (8) (2018) 5506–5519.
[28] J. Kwak, Y. Kim, J. Lee, S. Chong, DREAM: Dynamic resource and task allocation for energy minimization in mobile cloud systems, IEEE J. Sel. Areas Commun. 33 (12) (2015) 2510–2523.
[29] Y. Wang, M. Sheng, X. Wang, L. Wang, J. Li, Mobile-edge computing: Partial computation offloading using dynamic voltage scaling, IEEE Trans. Commun. 64 (10) (2016) 4268–4282.
[30] S. Khalili, O. Simeone, Inter-layer per-mobile optimization of cloud mobile computing: A message-passing approach, Trans. Emerg. Telecommun. Technol. 27 (6) (2016) 814–827.
[31] Y. Mao, J. Zhang, S.H. Song, K.B. Letaief, Power-delay tradeoff in multi-user mobile-edge computing systems, in: 2016 IEEE Global Communications Conference (GLOBECOM), 2016, pp. 1–6.
[32] M. Chen, B. Liang, M. Dong, Joint offloading and resource allocation for computation and communication in mobile cloud with computing access point, in: IEEE Conference on Computer Communications(INFOCOM), 2017, pp. 1–9.
[33] C. Wang, C. Liang, F.R. Yu, Q. Chen, L. Tang, Computation offloading and resource allocation in wireless cellular networks with mobile edge computing, IEEE Trans. Wirel. Commun. 16 (8) (2017) 4924–4938.
[34] X. Lyu, H. Tian, C. Sengul, P. Zhang, Multiuser joint task offloading and resource optimization in proximate clouds, IEEE Trans. Veh. Technol. 66 (4) (2017) 3435–3447.
[35] R. Yu, J. Ding, S. Maharjan, S. Gjessing, Y. Zhang, D.H.K. Tsang, Decentralized and optimal resource cooperation in geo-distributed mobile cloud computing, IEEE Trans. Emerg. Top. Comput. 6 (1) (2018) 72–84.
[36] Q. Zhang, B. Luo, W. Shi, A.M. Almoharib, Cloudsafe: Storing your digital asset in the cloud-based safe, Wayne State University, Detroit, USA, Tech. Rep, 2013.
[37] L. Tawalbeh, R.S. Al-Qassas, N.S. Darwazeh, Y. Jararweh, F. AlDosari, Secure and efficient cloud computing framework, in: Cloud and Autonomic Computing (ICCAC), 2015 International Conference on, IEEE, 2015, pp. 291–295.
[38] K. Fan, J. Wang, X. Wang, H. Li, Y. Yang, A secure and verifiable outsourced access control scheme in fog-cloud computing, Sensors 17 (7) (2017) 1695–1709.
[39] S. Almajali, H.B. Salameh, M. Ayyash, H. Elgala, A framework for efficient and secured mobility of iot devices in mobile edge computing, in: Fog and Mobile Edge Computing (FMEC), 2018 Third International Conference on, IEEE, 2018, pp. 58–62.
[40] T. Meng, K. Wolter, H. Wu, Q. Wang, A secure and cost-efficient offloading policy for mobile cloud computing against timing attacks, Pervasive Mob. Comput. 45 (2018) 4–18.
[41] L. Yang, J. Cao, H. Cheng, Y. Ji, Multi-user computation partitioning for latency sensitive mobile cloud applications, IEEE Trans. Comput. 64 (8) (2015) 2253–2266.
[42] X. Chen, Decentralized computation offloading game for mobile cloud computing, IEEE Trans. Parallel Distrib. Syst. 26 (4) (2015) 974–983.
[43] G. Iosifidis, L. Gao, J. Huang, L. Tassiulas, A double-auction mechanism for mobile data-offloading markets, IEEE/ACM Trans. Netw. 23 (5) (2015) 1634–1647.
[44] S. Deb, P. Monogioudis, Learning-based uplink interference management in 4g LTE cellular systems, IEEE/ACM Trans. Netw. 23 (2) (2015) 398–411.
[45] T.S. Rappaport, Wireless Communications: Principles and Practice, Horwood Publishing Limited Chichester, 2009, pp. 33–38.
[46] X. Lin, Y. Wang, Q. Xie, M. Pedram, Task scheduling with dynamic voltage and frequency scaling for energy minimization in the mobile cloud computing environment, IEEE Trans. Serv. Comput. 8 (2) (2015) 175–186.
[47] I. Elgendy, W. Zhang, C. Liu, C. Hsu, An efficient and secured framework for mobile cloud computing, IEEE Trans. Cloud Comput. (2018).
[48] J. Daemen, V. Rijmen (Eds.), The Design of Rijndael: AES - The Advanced Encryption Standard, Springer-Verlag, Berlin, Heidelberg, 2002.
[49] C. Paar, J. Pelzl, Understanding Cryptography: A Textbook for Students and Practitioners, first ed., Springer Publishing Company, Incorporated, 2009.

Ibrahim Elgendy received his M.Sc. degree from Computer Science Department, Faculty of Computers and Information, Menoufia University, Egypt, in 2016. He worked as a demonstrator and assistant lecturer in Faculty of Computers and Information, Menoufia University, Egypt since April 2012 till now. He is currently pursuing the Ph.D. degree from the School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China. His research interests include Cloud Computing, Mobile Edge Computing and distributed computing.

Weizhe Zhang is currently a professor in the School of Computer Science and Technology at Harbin Institute of Technology, China. His research interests are primarily in parallel computing, distributed computing, cloud and grid computing, and computer network. He has published more than 100 academic papers in journals, books, and conference proceedings. He is a senior member of the IEEE.

Yu-Chu Tian received the Ph.D. degree in computer and software engineering in 2009 from the University of Sydney, Sydney NSW, Australia, and the Ph.D. degree in industrial automation in 1993 from Zhejiang University, Hangzhou, China. He is currently a professor at the School of Electrical Engineering and Computer Science, Queensland University of Technology, Brisbane QLD, Australia. His research interests include big data computing, distributed computing, cloud computing, real-time computing, computer networks, and control systems.

Keqin Li is a SUNY Distinguished Professor of computer science in the State University of New York. He is also a Distinguished Professor of Chinese National Recruitment Program of Global Experts (1000 Plan) at Hunan University, China. He was an Intellectual Ventures endowed visiting chair professor at the National Laboratory for Information Science and Technology, Tsinghua University, Beijing, China, during 2011–2014. His current research interests include parallel computing and high-performance computing, distributed computing, energy-efficient computing and communication, heterogeneous computing systems, cloud computing, big data computing, CPU–GPU hybrid and cooperative computing, multicore computing, storage and le systems, wireless communication networks, sensor networks, peer-to-peer le sharing systems, mobile computing, service computing, Internet of things and cyber–physical systems. He has published over 520 journal articles, book chapters, and refereed conference papers, and has received several best paper awards. He is currently or has served on the editorial boards of *IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Computers, IEEE Transactions on Cloud Computing, IEEE Transactions on Services Computing, and IEEE Transactions on Sustainable Computing*. He is an IEEE Fellow.