Article

Multimedia Applications Processing and Computation Resource Allocation in MEC-Assisted SIoT Systems with DVS

Xianwei Li [1], **Guolong Chen** [1,2,*], **Liang Zhao** [3] and **Bo Wei** [4]

[1] School of Computer and Information Engineering, Bengbu University, Bengbu 233000, China; lixianwei163@163.com
[2] School of Information Engineering, Suzhou University, Suzhou 234000, China
[3] School of Computer Science, Shenyang Aerospace University, Shenyang 110000, China; lzhao@sau.edu.cn
[4] Graduate School of Fundamental Science and Engineering, Waseda University, Tokyo 169-0051, Japan; weibo0504@fuji.waseda.jp
* Correspondence: cglbox@sina.com

Abstract: Due to the advancements of information technologies and the Internet of Things (IoT), the number of distributed sensors and IoT devices in the social IoT (SIoT) systems is proliferating. This has led to various multimedia applications, face recognition and augmented reality (AR). These applications are computation-intensive and delay-sensitive and have become popular in our daily life. However, IoT devices are well-known for their constrained computational resources, which hinders the execution of these applications. Mobile edge computing (MEC) has appeared and been deemed a prospective paradigm to solve this issue. Migrating the applications of IoT devices to be executed in the edge cloud can not only provide computational resources to process these applications but also lower the transmission latency between the IoT devices and the edge cloud. In this paper, computation resource allocation and multimedia applications offloading in MEC-assisted SIoT systems are investigated. We aim to optimize the resource allocation and application offloading by jointly minimizing the execution latency of multimedia applications and the consumed energy of IoT devices. The studied problem is a formulation of the total computation overhead minimization problem by optimizing the computational resources in the edge servers. Besides, as the technology of dynamic voltage scaling (DVS) can offer more flexibility for the MEC system design, we incorporate it into the application offloading. Since the studied problem is a mixed-integer nonlinear programming (MINP) problem, an efficient method is proposed to address it. By comparing with the baseline schemes, the theoretic analysis and simulation results demonstrate that the proposed multimedia applications offloading method can improve the performances of MEC-assisted SIoT systems for the most part.

Keywords: multimedia applications; resource allocation; mobile edge computing

MSC: 90C26

check for updates

Citation: Li, X.; Chen, G.; Zhao, L.; Wei, B. Multimedia Applications Processing and Computation Resource Allocation in MEC-Assisted SIoT Systems with DVS. *Mathematics* **2022**, *10*, 1593. https://doi.org/10.3390/math10091593

Academic Editor: Ezequiel López-Rubio

Received: 7 April 2022
Accepted: 5 May 2022
Published: 7 May 2022

1. Introduction

Due to the advancements of information technologies [1–3], as well as the expansion of the IoT [4–7], the number of sensors and other IoT devices distributed by IoT users in the SIoT systems is growing explosively [8,9]. It is reported by Cisco that the number of the IoT devices and connections in the globe will be 13.5 billion by the year 2022, as can be observed from Figure 1 [10]. Accordingly, various multimedia applications, such as face recognition and augmented reality (AR), are appearing and have become popular in our daily life [11]. These multimedia applications are typically computation-intensive and need real-time processing [12]. However, the computation resources of these IoT nodes are insufficient enough to meet the demands of these applications [13,14].

Mobile cloud computing (MCC) was once regarded as an important paradigm to conquer the limitations of IoT devices, where the computation-intensive IoT applications can be migrated and processed in the distant public cloud [15]. However, one of the obvious shortcomings of this method is the high latency for the long transmission delay, which is not a suitable choice for the processing of latency-sensitive multimedia applications. Therefore, a promising paradigm, which is MEC, has been introduced [16–18]. By distributing the cloud resources near the edge of the wireless networks, MEC can not only provide computing resources to the IoT devices but also lower the latency from the transmission of the IoT devices to the edge cloud.

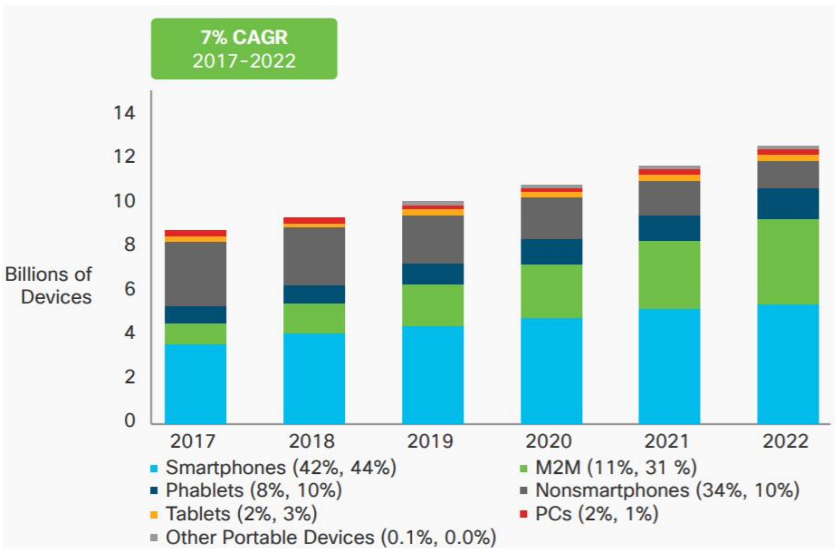


Figure 1. The growth of devices and connections worldwide [10].

Despite the advantages that the MEC has, how efficiently to allocate the computing resources of the MEC servers, such that both the application execution latency and consumed energy of IoT devices are minimized, is a challenging issue. From the viewpoint of the SIoT systems, designing efficient resources allocation algorithms can improve its safety, reliability and security, which ultimately contributes to the improvement of the reliability [19,20]. The DVS technology can vary the voltage and clock frequency of IoT devices according to the computation load such that the desired performance is provided [21]. By resorting to the DVS technique, the IoT devices cloud make adjustments to their computational speeds in order to reduce the application completion time and consumed energy. Therefore, combining the DVS technology with the application offloading can make the offloading strategy design more flexible. Although a significant amount of research attention has been paid to resource allocation and applications offloading in the MEC systems, joint consideration of the computational resources in the edge servers and adopting the DVS technology is not fully investigated.

In this paper, we make efforts to study computation resource allocation and multimedia applications offloading problems to optimize the performances in MEC-assisted SIoT systems. We formulate the studied problem as a latency and energy consumption minimization problem by optimizing the computational resources in the edge servers. Besides, as the DVS can offer more flexibility for the MEC system design, we incorporate it into the application offloading. Since the studied problem is an MINP problem, a method proven to be efficient is proposed to address it. Simulation results are presented to demonstrate the efficiency of the proposed multimedia applications offloading scheme by comparing it with the baseline schemes.

In summary, we demonstrate the main contributions in this paper as follows:

- The minimization problem of multimedia application execution latency and energy consumption of IoT devices is studied by the allocation of computing resources in the edge servers while adopting the DVS technology.

- The studied problem of latency and energy consumption is formulated. Due to the formulated problem being an MINP problem, an efficient multimedia applications offloading scheme is proposed, and the solution of it is obtained.
- Simulation results are performed to evaluate the efficacy of the proposed multimedia applications offloading scheme by comparing with the two baseline schemes. The theoretical analysis and simulation results indicate that the multimedia applications' offloading scheme proposed in this paper can perform better than the baseline methods, which can integrate the dependability aspects into the design of SIoT systems.

2. Related Work

Applications offloading, also known as data offloading, task offloading and computation offloading, and together with resource allocation, has been broadly studied in the current works. The allocated resources can be computing resources of MEC servers, radio resources, transmission power, and caching resources. The main objectives of them focus on the minimization of energy consumption, the application execution time, and both the application processing time and the consumed energy. We have referred to many of the relevant studies in the top renowned journals, which are included in the famous databases in the literature, such as IEEE Xplore and Elsevier. Based on the relevant studies, we discuss the related work from the following three perspectives.

(1) Energy consumption minimization.

In [22], Li et al. studied computation offloading problem for total energy consumption minimization by allocating transmission power in the dual connectivity (DC)-and nonorthogonal multiple access (NOMA)-assisted MEC systems. As the formulated problem is an MINLP problem, they proposed an iterative optimization method to obtain the solution. In [23], Wang et al. studied transmit power allocation and computing resources allocation for task offloading in order to obtain the minimization of the energy consumption of the mobile user equipment. In particular, they consider the imperfect channel state information (CSI). In [24], Liu et al. investigated the minimization of energy consumption of mobile devices in MCC systems under the constraints of transmission error rate and time delay. In [25], You et al. studied computing resource allocation of edge servers for mobile energy consumption minimization with computation latency constraint in MEC systems based on TDMA and OFDMA. In [26], Lyu et al. studied energy-efficient task admission for delay-sensitive applications in MEC systems. They formulated the total energy consumption minimization under the constraints of computational resources in the edge servers and latency. In [27], Zhao et al. studied the problem of energy consumption minimization by optimizing radio and computational resources of edge servers for smart mobile devices of the MEC system with multiple users. As the formulated problem is MINP, they proposed a method to obtain the optimal result. In [28], Zeng and Fodor investigated the problem of transmission energy consumption by allocating radio and computational resources jointly under a delay constraint. In [29], Meskar et al. studied commutation offloading in a MEC system to obtain the reduction of energy consumption. They used the competitive game to model the system, and a Gauss–Seidel method was introduced to determine the equilibrium. The optimization of resource allocation is not considered. In [14], Li et al. studied computation offloading for big data processing in the energy efficient large-scale IoT systems. For the research line of these previous studies, they either just paid attention to minimize the energy consumption or ignored the joint consideration of DVS technology and computation resource allocation.

(2) Application execution time minimization.

In [30], the problem of spectrum allocation and transmission power allocation for applications offloading is investigated to reduce the execution time, including computational time of MEC servers and transmission time of all mobile users. Because of the complexity of the optimized problem, they developed a reinforcement learning-based method to obtain the solution for the problem. In [31], Sun et al. studied the management of energy-aware mobility for MEC systems in the ultra-dense networks environment. They optimized the computation delay under the constraint of energy consumption by applying the Lyapunov optimization method. However, they did not consider resource allocations. In [32], Qian et al. studied the overall completing time reduction of the workload of smart terminal by formulating the optimization problem of computation resource allocation of edge servers. An efficient layered algorithm was proposed by them to obtain the optimal solution. In [33], Liu et al. studied a MEC system with two IoT devices offloading their tasks to an edge server so that the execution latency of IoT users can be minimized by considering the hybrid NOMA-OMA transmission. In [34], Chen et al. studied the minimization of compression execution time of multimedia applications in MEC systems. However, they only considered the optimization of radio and computation resources and ignored the adoption of DVS technology. As far as the research line of the aforementioned studies is concerned, their focus was concentrated on minimizing the application execution time, ignoring the joint consideration of DVS technology.

(3) Both the energy and application processing time minimization.

In [21], Wang et al. studied partial computation offloading by allocating the transmit power of mobile devices with DVS. They have two system design objectives, namely, the minimization of consumed energy and task execution time. In [35], Zhong et al. investigated the problem of task offloading by intelligently and efficiently manage resources in MEC for minimizing delay and energy consumption. They put forward a CL-ADMM framework that can reduce the delay and energy consumption effectively. In [36], Yang et al. studied task offloading for minimizing the energy and delay in the NOMA-enabled MEC systems with small cell networks. A hybrid genetic hill climbing (HGHC) algorithm is put forward to obtain the optimal solution to the formulated problem. In [37], Zhang et al. studied the tradeoff of energy and latency in the energy-aware MEC systems. A task offloading scheme was presented by the optimal allocation of the communication resource and using the DVS technology under the constraints of energy and delay. In our previous work [38], we studied bandwidth allocation for minimizing the system cost, which is measured by the task completion time and consumed energy of IoT devices, adopting the DVS technology. In [39], the authors studied transmission power and communication resources allocation for the balance of energy and latency task offloading in the full-duplex MEC systems. In [40], Zhu and Wen studied the allocation of transmission power, wireless resources, and computing resources in the edge servers to obtain the tradeoff of delay and energy. A computation task offloading policy was proposed by using the genetic algorithm. Although these previous studies considered the objectives of minimizing both energy and execution time, the DVS technology is overlooked.

From the above analysis, it is obvious that the allocation of computing resources in the edge cloud and the technology of DVS are not jointly considered in these previous works. Although the authors in [41] studied resource allocation for big data processing, their study is conducted in the geo-distributed federated public clouds. In addition, the joint objectives of the task completion time and energy consumed by IoT devices are not fully studied. We compare these previous works with the study of this paper, the results of which are listed in Table 1. From Table 1, it is clearly observed that the study in this paper can overcome the shortcomings that exist in the previous works and also provide a complement to them.

Table 1. Comparison.

Reference	DVS	Application Execution Time	Energy Consumption	Computation Resources
Chen et al. [16]	No	No	Yes	No
Wang et al. [21]	Yes	Yes	Yes	No
Wang et al. [30]	No	No	No	No
Zhong et al. [35]	No	Yes	Yes	No
Li et al. [22]	No	Yes	Yes	No
Wang et al. [23]	No	No	Yes	Yes
Liu et al. [33]	No	No	Yes	No

Table 1. *Cont.*

Reference	DVS	Application Execution Time	Energy Consumption	Computation Resources
Li et al. [14]	No	No	Yes	No
Li et al. [38]	Yes	Yes	Yes	No
Yang et al. [36]	No	Yes	Yes	No
Liu et al. [24]	No	Yes	Yes	Yes
Lyu et al. [26]	No	Yes	Yes	Yes
You et al. [25]	No	Yes	Yes	Yes
Zhao et al. [27]	No	Yes	Yes	Yes
Kabir et al. [39]	No	Yes	Yes	No
Zhang et al. [37]	Yes	Yes	Yes	No
Sun et al. [31]	No	Yes	Yes	No
Meskar et al. [29]	No	Yes	Yes	No
Zeng et al. [28]	No	Yes	Yes	No
Zhu et al. [40]	No	Yes	Yes	Yes
This Study	Yes	Yes	Yes	Yes

3. System Architecture

The system architecture is introduced in this section. Suppose that there is a MEC-assisted SIoT system with N IoT devices and one base station (BS), as Figure 2 shows. Each IoT device has a multimedia application that needs a lot of computation resources to be computed. Besides, an edge cloud is deployed near the BS to provide computation resources. As far as the IoT device i is concerned, $i \in \{1, 2, 3, ..., N\}$, its application can be represented by $I_i = (D_i, C_i)$, where D_i is the data size, and C_i is the needed number of CPU cycles for computing this application. The user of the IoT device i can use the methods in [42] to obtain the information of D_i and C_i. The IoT devices will make offloading decisions on whether to offload and process their applications in the edge servers or not. Let O_i denote the application offloading decision made by IoT device i. If the IoT device i offloads its application via BS to the edge cloud, $O_i = 1$, otherwise, $O_i = 0$. It should be noted that the system architecture of this paper can be utilized in numerous real-life cases. For example, the IoT devices may correspond to networked cameras of the surveillance monitoring systems, whose sensed video data require real-time execution, and the multimedia applications can face recognition applications.

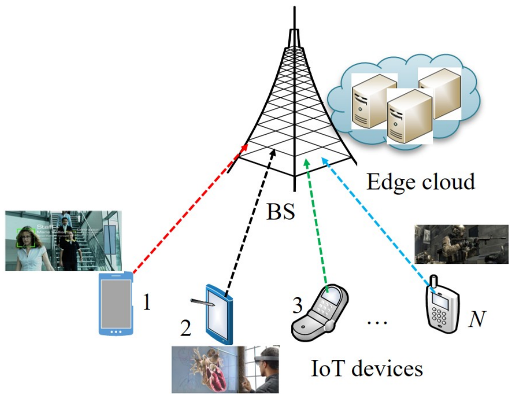


Figure 2. MEC-assisted social IoT system.

3.1. Local Computation Model

In this model, the application of the IoT device i is computed locally using its resources. We use f_i^l to denote the computing capacity in cycles per second. For IoT device i, the corresponding application execution time and consumed energy can be, respectively, denoted as [16]

$$t_{i,l} = \frac{C_i}{f_i^l} \tag{1}$$

$$e_{i,l} = P_{i,l} t_{i,l} \tag{2}$$

where $P_{i,l}$ is the consumed consumption for one CPU cycle.

The CPU power consumption per cycle is [16]

$$P_{i,l} = k_i (f_i^l)^3 \tag{3}$$

where k_i is a constant parameter, and its value relies on the chip architecture [37].

Therefore,

$$e_{i,l} = k_i C_i (f_i^l)^2 \tag{4}$$

Based on Equations (1) and (4), the local computation cost in the measure of the application execution time and the consumed energy is denoted as

$$\begin{aligned} G_i &= \lambda_{i,t} t_{i,l} + \lambda_{i,e} e_{i,l} \\ &= \lambda_{i,t} \frac{C_i}{f_i^l} + \lambda_{i,e} k_i C_i (f_i^l)^2 \end{aligned} \tag{5}$$

where $\lambda_{i,t}$ and $\lambda_{i,e} \in [0,1]$ represent the weight coefficient of the application processing time and consumed energy, respectively. If $\lambda_{i,t} > \lambda_{i,e}$, the IoT user puts more weight on processing time. Otherwise, if $\lambda_{i,t} < \lambda_{i,e}$, the IoT user puts more weight on the consumed energy such that its battery energy can be saved.

3.2. Edge Cloud Model

For the edge cloud model, IoT device i will offload and process its application in the edge cloud via wireless access, such as a macrocell BS or a WIFI access point.

For IoT device i, the uplink transmission data rate is

$$r_i = w_i \log_2 \left(1 + \frac{q_i h_i}{\varpi_0}\right) \tag{6}$$

where q_i is the transmit power, w_i is the channel bandwidth, h_i is the channel gain, and ϖ_0 is the background noise power.

Then, the consumed time of IoT device i in the uplink is

$$t_{i,t} = \frac{D_i}{r_i} = \frac{D_i}{w_i \log_2 (1 + \frac{q_i h_i}{\varpi_0})} \tag{7}$$

Accordingly, the consumed energy in the uplink is

$$e_{i,o} = q_i t_{i,t} = q_i \frac{D_i}{w_i \log_2 (1 + \frac{q_i h_i}{\varpi_0})} \tag{8}$$

After the applications of IoT devices are migrated to the edge cloud, the computing resources will be allocated to process them. For the IoT device i, its application processing time is

$$t_{i,e} = \frac{C_i}{f_i^e} \tag{9}$$

Based on Equations (7) and (9), the application offloading time is

$$t_{i,o} = t_{i,t} + t_{i,e} = \frac{D_i}{w_i \log_2 (1 + \frac{q_i h_i}{\varpi_0})} + \frac{C_i}{f_i^e} \tag{10}$$

Therefore, according to the Equations (8) and (10), the computation cost of application offloading for mobile device i is

$$\begin{aligned} H_i &= \lambda_{i,t} t_{i,o} + \lambda_{i,e} e_{i,o} \\ &= \lambda_{i,t} \left(\frac{D_i}{w_i \log_2(1 + \frac{q_i h_i}{\varpi_0})} + \frac{C_i}{f_i^e} \right) \\ &+ \lambda_{i,e} q_i \frac{D_i}{w_i \log_2(1 + \frac{q_i h_i}{\varpi_0})} \end{aligned} \tag{11}$$

3.3. Problem Formulation

The optimization problem is formulated by minimization of the computation cost for the local computation and edge cloud computation of IoT devices, which is

P:

$$\begin{aligned} &\min_{f_i^l, f_i^e, O_i} \sum_{i=1}^{N} Z_i \\ &\text{s.t.} \quad 0 < f_i^l \le f_{i,m}^l \\ &\qquad \sum_{i=1}^{N} f_i^e \le F \\ &\qquad O_i \in \{0, 1\} \end{aligned} \tag{12}$$

where Z_i is denoted as

$$\begin{aligned} Z_i &= (1 - O_i) G_i + O_i H_i \\ &= (1 - O_i)[\lambda_{i,t} t_{i,l} + \lambda_{i,e} e_{i,l}] \\ &+ O_i[\lambda_{i,t} t_{i,o} + \lambda_{i,e} e_{i,o}] \end{aligned} \tag{13}$$

where G_i and H_i are shown in the Equations (5) and (11), respectively.

In the above problem, constraint one is the processing capacity of the IoT device i, constraint two is the computation resources constraints of edge servers, and constraint three means the offloading decision that the IoT device i will make.

For convenient investigation, the notations are summarized in Table 2.

Remark 1. *Problem* ***P*** *is a typical MINP problem, the function of which is non-convex and notoriously hard to be solved [43]. This is due to the fact that the application offloading decision is binary, but the computation resource allocation is continuous. We can apply the Alternating Direction Method of Multipliers (ADMM), dandelion algorithm (DA) [44], Bat algorithm [45], and branch-and-bound methods to solve the MINP problem. However, the time-complexity is prohibitive [26]. The authors in [46] solved the MINP problem using the algorithms of benders' decomposition, admm, dinkelbach, and branch-and-bound, and compared the performances of these algorithms. In the following section, a method is proposed to obtain the optimal solution.*

Table 2. A summary of notations.

Notation	Description
f_i^l	The processing rate of the IoT device i
$t_{i,l}$	The local processing time of the IoT device i
$e_{i,l}$	The local energy consumption of the IoT device i
$t_{i,e}$	The transmission time of the IoT device i
$e_{i,o}$	The transmission energy consumption of the IoT device i
D_i	The application i's data size

Table 2. *Cont.*

Notation	Description
C_i	The needed CPU cycles to process application of IoT device i
$\lambda_{i,t}$	The weighting parameter of application processing time
$\lambda_{i,e}$	The weighting parameter of energy consumption
w_i	The channel bandwidth
q_i	The transmit power of the IoT device i
h_i	The channel gain
ϖ_0	The background noise power
r_i	The uplink transmission rate for the IoT device i
O_i	The application offloading decision that make by the IoT device i
f_i^e	The allocated computational resources in the edge cloud to device i
$f_{i,m}$	The maximum processing rate of the IoT device i
F	The total computational resources in the edge servers

4. Solution Method

In this section, the proposed solution method for the formulated application execution time and consumed energy minimization problem is introduced.

4.1. Solution Method

Before presenting the solution method, a lemma is firstly introduced, the proof of which is shown in [47]. Based on this lemma, the proposed solution method is introduced.

Lemma 1. *The following result is always true*

$$\sup_{x,y} f(x,y) = \sup_{x} \tilde{f}(x),$$

where $\tilde{f}(x) = sup_y f(x,y)$.

Lemma 1 shows that when trying to minimize a function, some variables can firstly be chosen to be minimized, and then the left ones are to be minimized.

Based on **Lemma 1**, Problem **P** can be solved by maximizing over f_i^l, and f_i^e sequentially. Therefore, the problem **P** can be solved from the solution of the below three subproblems:

(1) Local computation problem;

(2) Edge cloud computation problem;

(3) The offloading decision problem.

4.2. Local Computation Problem

When $O_i = 0$, the users of IoT devices will accomplish their applications on their own mobile devices. In this case, problem **P** becomes the following problem:

P1:

$$\begin{aligned} &\min_{f_i^l} G_i(f_i^l) = \lambda_{i,t}\frac{C_i}{f_i^l} + \lambda_{i,e}k_iC_i(f_i^l)^2 \\ &\text{s.t.} \quad 0 < f_i^l \le f_{i,m} \end{aligned} \tag{14}$$

From $\frac{\partial^2 G_i}{\partial (f_i^l)^2} > 0$, we know that $G_i(f_i^l)$ is convex. Hence, from the first derivative of $G_i(f_i^l)$ with respect to f_i^l,

$$\frac{\partial G_i}{\partial f_i^l} = -\frac{\lambda_{i,t}C_i}{(f_i^l)^2} + 2\lambda_{i,e}k_if_i^lC_i = 0 \tag{15}$$

From Equation (15), the following optimal solution is obtained.

$$f_i^{l^*} = \sqrt[3]{\frac{\lambda_{i,t}}{2k_i\lambda_{i,e}}} \tag{16}$$

It is obviously observed that $G_i(f_i^l)$ monotonously increases when $f_i^l > f_i^{l*}$ and monotonously increases when $f_i^l < f_i^{l*}$.

Therefore, the optimal solution for problem P1 is

$$G_i(f_i^l) = \begin{cases} G_i(f_{i,m}), f_i^{l*} \geq f_{i,m} \\ G_i(f_i^{l*}), f_i^{l*} < f_{i,m} \end{cases} \tag{17}$$

4.3. Edge Cloud Computation Problem

If the users of IoT devices make decisions to offload and process their applications in the edge cloud, problem **P** becomes the following problem:
P2:

$$\begin{gathered} \min_{f_i^e} \sum_{i=1}^{N} H_i \\ \sum_{i=1}^{N} f_i^e \leq F \\ f_i^e > 0 \end{gathered} \tag{18}$$

The objective function in the problem **P2** is easily checked as an obvious convex function. Hence, the Lagrangian function for problem **P2** is

$$\begin{aligned} L_i(f_i^e, \mu) = \lambda_{i,t}(&\frac{D_i}{w_i \log_2(1 + \frac{q_i h_i}{\varpi_0})} \\ &+ \frac{C_i}{f_i^e}) + \lambda_{i,e} q_i \frac{D_i}{w_i \log_2(1 + \frac{q_i h_i}{\varpi_0})} \\ &+ \mu(\sum_{i=1}^{N} f_i^e - F) \end{aligned} \tag{19}$$

where $\mu \geq 0$ denotes the Lagrangian multiplier.

Based on the KKT conditions [47], we have the following results.

$$\frac{\partial L_i}{\partial f_i^e} = -\frac{C_i \lambda_{i,t}}{r_i (f_i^e)^2} + \mu = 0 \tag{20}$$

$$\frac{\partial L_i}{\partial \mu} = \sum_{i=1}^{N} f_i^e - F = 0 \tag{21}$$

$$\mu \geq 0 \tag{22}$$

From Equation (20), it is evident that $\mu > 0$. Therefore, we obtain

$$f_i^e = \sqrt{\frac{C_i \lambda_{i,t}}{r_i \mu}} \tag{23}$$

Substituting Equation (23) back into Equation (21), we obtain

$$\mu = (\frac{\sum_{i=1}^{N}\sqrt{\frac{C_i\lambda_{i,t}}{r_i}}}{F})^2 \tag{24}$$

Substituting Equation (24) back into Equation (23), we obtain the optimal computational resource allocation of edge servers, which is denoted as

$$f_i^e = F\frac{\sqrt{\frac{C_i\lambda_{i,t}}{r_i}}}{\sum_{i=1}^{N}\sqrt{\frac{C_i\lambda_{i,t}}{r_i}}} \tag{25}$$

By solving the above two subproblems, we can obtain the computation cost in the two computation models.

4.4. Application Offloading Decision

In this part, the proposed efficient application offloading algorithm will be introduced, which is motivated by the work of [37]. The users of IoT devices will make decisions to offload and process their applications in the edge cloud if the computation cost of the edge cloud model is not higher than that of the local model [16]. The user of the IoT device *i* will make an application offloading decision by comparing the computation cost of the local and edge cloud computation,

$$O_i = \begin{cases} 1, G_i \geq H_i \\ 0, G_i < H_i \end{cases} \tag{26}$$

After the minimum computation cost of each IoT device is obtained, the total minimum system computation cost is denoted as

$$Z^* = \sum_{i=1}^{N}((1-O_i)G_i^* + O_iH_i^*) \tag{27}$$

Accordingly, the application offloading decisions of IoT devices in the MEC systems are made according to Algorithm 1.

Algorithm 1 Computational Resource Allocation and Applications Offloading Algorithm

```
Input:
 1: $N$ applications;
Output:
 2: The application offloading decision made by each user of IoT device and the system
    computation overhead;
 3: Obtain the optimal resource allocation of each IoT device by solving problem P1;
 4: Based on Equation (16), IoT devices make adjustments to their voltage and clock
    frequency to obtain the adaptive CPU frequency according to the weight coefficient
    values by applying the DVS technology;
 5: Obtain the optimized local computation cost of each IoT device based on Equation (17);
 6: Calculate the allocated computational resource of each IoT device by solving problem
    P2;
 7: Obtain the optimal computation cost of the edge cloud computation model from
    Equation (11);
 8: if $G_i \geq H_i$ then
 9:     $O_i = 1$;
10: else
11:     $O_i = 0$;
12: end if
```

5. Simulation Results

The proposed computational resource allocation and application offloading algorithm is evaluated in this section by simulation results. The proposed application offloading algorithm is compared with the following two benchmark algorithms:

Local Computing (LC) Algorithm: In this algorithm, all the IoT devices do not take the offloading strategy. They compute their applications by using the computing resources of their devices.

Edge Cloud (EC) Algorithm: In this algorithm, there is no local computation. All the IoT devices take the offloading strategy, offload and process their applications in the edge cloud.

5.1. Parameter Setting

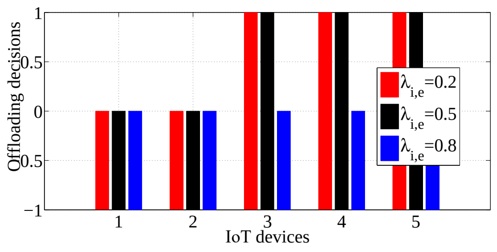
Suppose that a MEC-assisted IoT system exists with $N = 5$ IoT devices, randomly distributed over a region, with a BS being placed at the center of this region. There are servers located near the BS, whose computation capacity F is set as 20 GHz, and the maximum computation capacity for the IoT device i is $f_{i,m} = 1$ GHz. Each IoT device has one multimedia application which needs computation resources to be executed. The data sizes of multimedia applications have a uniform distribution in [0.42, 4.2] Mb, which corresponds to the face recognition application. The requirement of the number of CPU cycles of these applications have a uniform distribution in [0.1, 1] $\times 10^9$ cycles. The transmission power of each IoT device is set as 1 W. The allocated bandwidth for each IoT device is 1.6 MHz. The channel gain is $h_i = 2.6 \times 10^{-7}$. The background noise power is set to be 10^{-7} W. The weight coefficient coefficient values $\lambda_{i,t}$ and $\lambda_{i,e}$ are randomly chosen from $\{0.2, 0.5, 0.8\}$ and $\lambda_{i,t} + \lambda_{i,e} = 1$. The main simulation parameter values are listed in Table 3. These parameter values are set according to [16,26].

Table 3. Simulation parameter values.

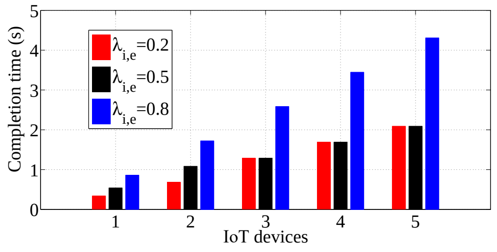
Parameters	Values
The number of IoT devices N	5
The bandwidth allocation w_i	1.6 MHz
The data sizes of multimedia applications D_i	[0.42, 4.2] Mb
The transmission power q_i	1 W
The needed computation resources C_i	$[0, 1] \times 10^9$ cycles
k_i	10^{-26}
The channel gain h_i	2.6×10^{-7}
The maximum processing rate $f_{i,m}$	1 GHz
The background noise power ϖ_0	10^{-7} W
The computation capacity of the edge cloud F	20 GHz

5.2. Effect of Weight Coefficient Values

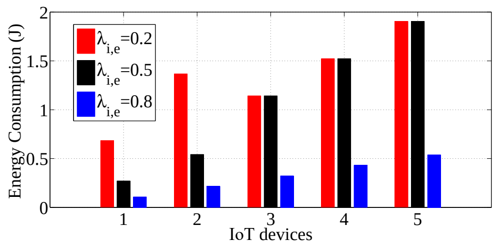
In this section, the effect of the values of weight parameters on the offloading decisions, application completion time, and consumed energy of IoT devices are analyzed. The values of $\lambda_{i,e}$ are set to be 0.2, 0.5, and 0.8. Figure 3a shows the effect of weight parameter values on the application offloading decisions for IoT users. It can be observed from Figure 3a that as the data sizes of their applications are small, IoT users 1 and 2 compute their applications by their own devices under different values. However, for the IoT users 3, 4, and 5, as their applications have larger data sizes, they compute their applications by using the computation resources of the edge cloud under the values of 0.2 and 0.5. Furthermore, all the IoT users make the decisions on computing their applications on their own devices when the value is 0.8. This is due to the reason that offloading these applications to the edge cloud will consume more energy.



(**a**) Offloading decisions



(**b**) Application completion time



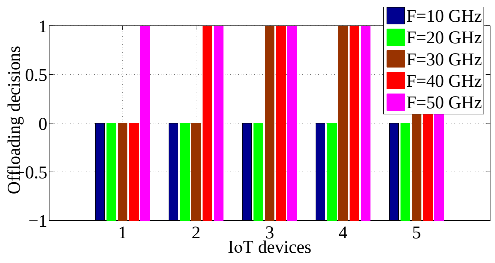
(**c**) Computation Cost

Figure 3. The effect of the weight coefficient values.

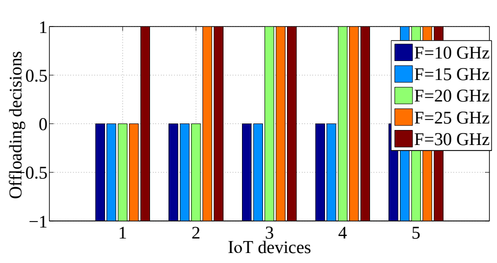
Figure 3b,c show the effect of weight coefficient values on the application completion time and the consumed energy, respectively. It can be obviously observed from Figure 3b that the completion time of the applications of IoT devices increases with the value of $\lambda_{i,e}$ increasing. The IoT devices, whose applications are of larger data sizes, will cost more time to complete their applications. Meanwhile, we have the observation from Figure 3c that the energy consumed by the IoT devices will decrease if the weight coefficient value $\lambda_{i,e}$ increases. This is because with the weight coefficient increasing value of $\lambda_{i,e}$, IoT users pay more attention to the factor of energy consumption in this scenario. Figure 3b,c also indicates that we could tune the values of the weight coefficient in different scenarios to meet the requirements of the MEC-assisted SIoT system.

5.3. Effect of the Edge Cloud Capacity

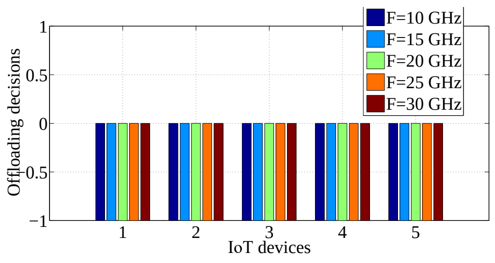
The effect of the edge cloud capacity on the offloading decisions of IoT users and the computation cost is analyzed in this section. The simulation results are shown in Figures 4–7. Figure 4a–c depict how the edge cloud capacity affects the offloading decisions that IoT users will make versus the weight coefficient values. From Figure 4a, we observe that when $\lambda_{i,e} = 0.2, \lambda_{i,t} = 0.8$, IoT devices 1 and 2 do not offload their applications even if the capacity of the edge cloud increases. This is because these IoT devices pay more attention to the completion time, and local processing will cost less execution time for them. IoT devices 3, 4, and 5 will offload applications with the capacity of the edge cloud increasing. Through Figure 4b, we find that the users of IoT devices would like to offload their applications if the edge cloud capacity increases. In particular, all the users of IoT devices offload their applications when $F = 30$ GHz. When $\lambda_{i,e} = 0.8, \lambda_{i,t} = 0.2$, we see from Figure 4c that all the IoT devices are to complete their applications locally. This is due to the reason that more attention is paid to the energy consumption in this scenario, and offloading applications to the edge cloud may cost more energy than the consumed energy in the local computing model. Therefore, IoT users would like to execute their applications locally. Figure 4a–c demonstrate that the offloading decisions that the IoT users make depend not only on the chosen weight coefficient values, but also the capacity of the edge cloud.



(**a**) $\lambda_{i,e} = 0.2, \lambda_{i,t} = 0.8$



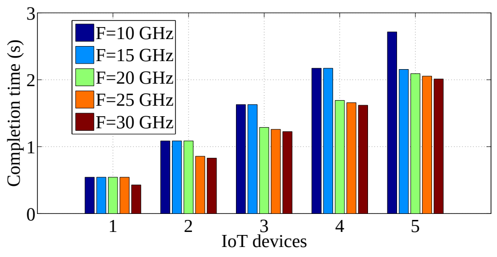
(**b**) $\lambda_{i,e} = 0.5, \lambda_{i,t} = 0.5$



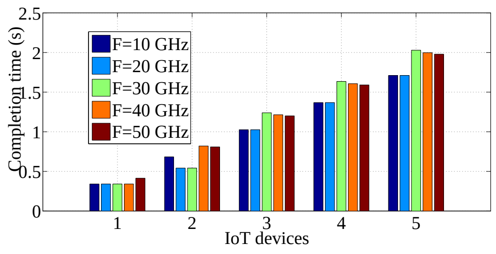
(**c**) $\lambda_{i,e} = 0.8, \lambda_{i,t} = 0.2$

Figure 4. The effect of the edge cloud capacity on the offloading decisions.

Figure 5a,b depict how the edge cloud capacity affects the application completion time under different values of weight coefficient. We see that the application completion time for each IoT device will decrease if the edge cloud capacity increases. By comparing Figure 5a,b, we find that the application completion time for each IoT device in Figure 5b is smaller than that in the Figure 5a under fixed edge cloud capacity. This is owing to the fact that the coefficient value of $\lambda_{i,t}$ is smaller in Figure 5a.



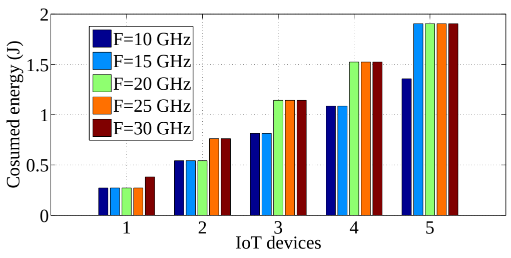
(**a**) $\lambda_{i,e} = 0.5, \lambda_{i,t} = 0.5$



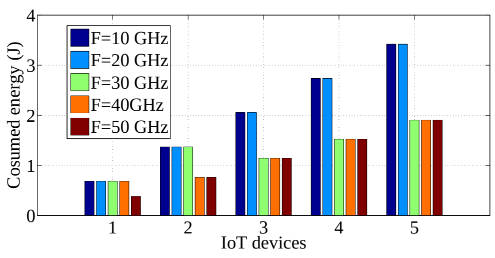
(**b**) $\lambda_{i,e} = 0.2, \lambda_{i,t} = 0.8$

Figure 5. The effect of the edge cloud capacity on the application completion time.

Figure 6a,b depict how the edge cloud capacity affects the consumed energy under different values of weight coefficient. By comparing the two figures, we observe that the consumed energy for each IoT device in Figure 6a is lower than that in Figure 6b. This is owing to the fact that the coefficient value of $\lambda_{i,e}$ is bigger in Figure 6a.



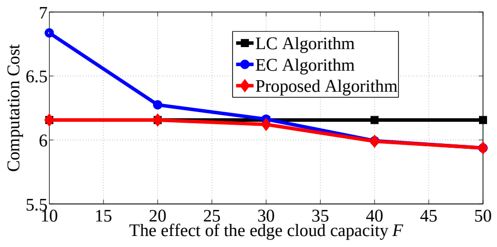
(**a**) $\lambda_{i,e} = 0.5, \lambda_{i,t} = 0.5$



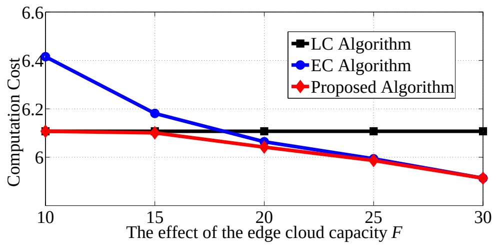
(**b**) $\lambda_{i,e} = 0.2, \lambda_{i,t} = 0.8$

Figure 6. The effect of the edge cloud capacity on the consumed energy.

Figure 7a,b show how the edge cloud capacity impacts the computation cost with different values of weight parameters. The proposed algorithm is compared with two baseline algorithms. It is obviously shown in these figures that the computation costs under the **EC Algorithm** and the **Proposed Algorithm** decrease with the increase in the edge cloud capacity. As the edge cloud capacity does not affect the local computation cost, the computation cost in the **LC Algorithm** has the constant value. From Figure 7a,b, as anticipated, the **Proposed Algorithm** can achieve less computation cost compared with the **LC Algorithm** and the **EC Algorithm**. By comparing Figure 7a,b, it can also be obviously observed that the computation cost is lower when $\lambda_{i,e} = \lambda_{i,t} = 0.5$. Moreover, the two figures show that the computation cost in the **LC Algorithm** is lower than that in the **EC Algorithm** until the edge cloud capacity reaches a threshold value. When $\lambda_{i,e} = 0.5, \lambda_{i,t} = 0.5$, the threshold value is comparatively smaller.



(**a**) $\lambda_{i,e} = 0.2, \lambda_{i,t} = 0.8$



(**b**) $\lambda_{i,e} = 0.5, \lambda_{i,t} = 0.5$

Figure 7. The effect of the edge cloud capacity on the computation cost.

There are several factors that could potentially affect the simulation results. For example, the weight coefficient values of $\lambda_{i,e}$ and $\lambda_{i,t}$ should be selected properly. In addition, the computing resources provided by the edge cloud provider should be adjusted according to data sizes of applications and the number of IoT users. Otherwise, there will be biases in the experimental results.

6. Conclusions

This paper has investigated resource allocation and multimedia applications offloading in a MEC-assisted SIoT system. Specifically, we analyzed the allocation of computation resources and the adoption of the DVS technology to minimize the computation cost in the measure of the application completion time and consumed energy for the processing of multimedia applications of IoT devices. We formulated the application completion time and consumed energy of IoT devices as an optimization problem. As the formulated problem is MINP, it is non-convex and NP-hard. The problem is decomposed into three sub-problems, and the solution for each one has been analyzed and solved. Theoretic analysis and simulation results have demonstrated the performance of the proposed algorithm by comparing the two baseline methods. The proposed method is beneficial for the efficient allocation of computation resource and application offloading in the MEC-assisted SIoT system. The experiment results highlight that the weight coefficient values and the edge cloud capacity impact the offloading decisions IoT users dramatically. In contrast to the two benchmark algorithms, the proposed algorithm reveals its advantages in terms of application completion time, consumed energy, and computation cost.

Several research issues should be studied in the future works. First, we will extend our study to consider the allocation of the caching resources [48] and apply the deep learning algorithms to solve the objective problem [49]. Second, as the backscatter communication (BackCom) can improve the transmission rate [50], we will combine the BackCom with computation offloading in MEC. Third, the Software Defined Network (SDN) technique will be integrated into the application offloading [51]. In addition, the security issue in the MEC-assisted SIoT system is also an interesting research topic [52].

Author Contributions: Conceptualization, X.L.; Funding acquisition, G.C. and B.W.; Investigation, X.L. and B.W.; Methodology, G.C.; Project administration, G.C.; Software, X.L.; Supervision, G.C. and L.Z.; Validation, L.Z. and B.W.; Writing–original draft, X.L.; Writing–review & editing, L.Z. and B.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Start up funds for the scientific research of high level talents of Bengbu University (BBXY2020KYQD02), Funding project for the cultivation of outstanding talents in Colleges and Universities (gxyqZD2021135), Key research and development projects in Anhui Province (202004a05020043), High level scientific research and Cultivation Project of Benbu University (2021pyxm05), Peak discipline construction of Bengbu University (2021GFXK03), JSPS KAKENHI Grant Number 20K14740 and JST, PRESTO Grant Number JPMJPR21PB, Japan, and Scientific Research and Development Fund of Suzhou University (2021fzjj29).

Acknowledgments: The authors would like to pay thanks to the anonymous reviewers for their valuable comments and suggestions, which greatly improve the quality of this paper.

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

Abbreviations

The following abbreviations are used in this manuscript:

MEC	Mobile Edge Computing
MCC	Mobile Cloud Computing
SIoT	Social Internet of Things
DVS	Dynamic voltage scaling
MINP	Mixed-integer nonlinear programming

References

1. Gong, Y.; Zhang, L.; Liu, R.P.; Yu, K.; Srivastava, G. Nonlinear mimo for industrial internet of things in cyberphysical systems. *IEEE Trans. Ind. Inform.* **2021**, *8*, 5533–5541. [CrossRef]
2. Ding, F.; Zhu, G.; Alazab, M.; Li, X.; Yu, K. Deep-learning-empowered digital forensics for edge consumer electronics in 5g hetnets. *IEEE Consum. Electron. Mag.* **2022**, *11*, 42–50. [CrossRef]
3. Guo, Z.; Yu, K.; Li, Y.; Srivastava, G.; Lin, J.C.-W. Deep Learning-Embedded Social Internet of Things for Ambiguity-Aware Social Recommendations. *IEEE Trans. Netw. Sci. Eng.* **2021**, 1. [CrossRef]
4. Tan, L.; Yu, K.; Ming, F.; Chen, X.; Srivastava, G. Secure and resilient artificial intelligence of things: A honeynet approach for threat detection and situational awareness. *IEEE Consum. Electron. Mag.* **2021**, *11*, 69–78. [CrossRef]
5. Li, H.; Yu, K.; Liu, B.; Feng, C.; Qin, Z.; Srivastava, G. An efficient ciphertext-policy weighted attribute based encryption for the internet of health things. *IEEE J. Biomed. Health Inform.* **2021**, *26*, 1949–1960. [CrossRef]
6. Zhao, L.; Zheng, T.; Lin, M.; Hawbani, A.; Shang, J.; Fan, C. SPIDER: A Social Computing Inspired Predictive Routing Scheme for Softwarized Vehicular Networks. *IEEE Trans. Intell. Transp.* **2021**, 1–12. [CrossRef]
7. Zhen, L.; Zhang, Y.; Yu, K.; Kumar, N.; Barnawi, A.; Xie, Y. Early collision detection for massive random access in satellite-based internet of things. *IEEE Trans. Veh. Technol.* **2021**, *70*, 5184–5189. [CrossRef]
8. Zhao, L.; Wang, C.; Li, W.; Zhao, K.; Tarchi, D.; Wan, S.; Kumar, N. INTERLINK: A Digital Twin-Assisted Storage Strategy for Satellite-Terrestrial Networks. *IEEE Trans. Aerosp. Electron. Syst.* **2021**, 1. [CrossRef]
9. Feng, C.; Liu, B.; Guo, Z.; Yu, K.; Qin, Z.; Choo, K.-K.R. Blockchain-based cross-domain authentication for intelligent 5g-enabled internet of drones. *IEEE Internet Things* **2022**, *9*, 6224–6238. [CrossRef]
10. Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update 2017–2022 White Paper. Available online: http://media.mediapost.com/uploads/CiscoForecast.pdf (accessed on 6 April 2022).
11. Sun, Y.; Liu, J.; Yu, K.; Alazab, M.; Lin, K. Pmrss: Privacy-preserving medical record searching scheme for intelligent diagnosis in iot healthcare. *IEEE Trans. Veh. Technol.* **2022**, *18*, 1981–1990. [CrossRef]
12. Shang, W.-L.; Chen, J.; Bi, H.; Sui, Y.; Chen, Y.; Yu, H. Impacts of covid-19 pandemic on user behaviors and environmental benefits of bike sharing: A big-data analysis. *Appl. Energy* **2022**, *285*, 116429–116429. [CrossRef]
13. Sheng, Z.; Wang, H.; Yin, C.; Hu, X.; Yang, S.; Leung, V.C.M. Lightweight management of resource-constrained sensor devices in internet of things. *IEEE Internet Things* **2015**, *2*, 402–411. [CrossRef]
14. Li, G.; He, J.; Peng, S.; Jia, W.; Wang, C.; Niu, J.; Yu, S. Energy efficient data collection in large-scale internet of things via computation offloading. *IEEE Internet Things* **2019**, *35*, 4176–4187. [CrossRef]
15. Tan, L.; Yu, K.; Shi, N.; Yang, C.; Wei, W.; Lu, H. Towards secure and privacy-preserving data sharing for COVID-19 medical records: A blockchain-empowered approach. *IEEE Trans. Netw. Sci. Eng.* **2022**, *1*, 271–281. [CrossRef]
16. 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]

17. Shi, W.; Cao, J.; Zhang, Q.; Li, Y.; Xu, L. Edge computing: Vision and challenges. *IEEE Internet Things* **2016**, *5*, 637–646. [CrossRef]
18. Liu, L.; Feng, J.; Pei, Q.; Chen, C.; Ming, Y.; Shang, B.; Dong, M. Blockchain-enabled secure data sharing scheme in mobile-edge computing: An asynchronous advantage actorcritic learning approach. *IEEE Internet Things* **2021**, *4*, 2342–2353. [CrossRef]
19. Mahmoud, R.; Yousuf, T.; Aloul, F.; Zualkernan, I. Internet of things (IoT) security: Current status, challenges and prospective measures. In Proceedings of the 10th International Conference for Internet Technology and Secured Transactions (ICITST 2015), London, UK, 14–16 December 2015; pp. 336–341.
20. Bures, M.; Klima, M.; Rechtberger, V.; Ahmed, B.S.; Hindy, H.; Bellekens, X. Review of Specific Features and Challenges in the Current Internet of Things Systems Impacting Their Security and Reliability. In Proceedings of the Trends and Applications in Information Systems and Technologies, Terceira Island, Azores, Portugal, 30 March–2 April 2021; pp. 546–556.
21. Wang, Y.; Sheng, M.; Wang, X.; Wang, L.; Li, J. Mobile-edge computing: Partial computation offloading using dynamic voltage scaling. *IEEE Trans. Commun.* **2016**, *10*, 4268–4282. [CrossRef]
22. Li, C.; Wang, H.; Song, R. Intelligent offloading for noma-assisted mec via dual connectivity. *IEEE Internet Things* **2021**, *4*, 2802–2813. [CrossRef]
23. Wang, J.; Feng, D.; Zhang, S.; Liu, A.; Xia, X.-G. Joint computation offloading and resource allocation for mec-enabled iot systems with imperfect csi. *IEEE Internet Things* **2021**, *5*, 3462–3475. [CrossRef]
24. Liu, K.; Peng, J.; Li, H.; Zhang, X.; Liu, W. Multi-device task offloading with time-constraints for energy efficiency in mobile cloud computing. *Perform. Eval. Rev.* **2016**, *64*, 1–14. [CrossRef]
25. You, C.; Huang, K.; Chae, H.; Kim, B.-H. Energy-Efficient Resource Allocation for Mobile-Edge Computation Offloading. *IEEE Wirel. Commun.* **2017**, *16*, 1397–1411. [CrossRef]
26. Lyu, X.; Tian, H.; Ni, W.; Zhang, Y.; Zhang, P.; Liu, R.P. Energy-efficient admission of delay-sensitive tasks for mobile edge computing. *IEEE Trans. Commun.* **2018**, *6*, 2603–2616. [CrossRef]
27. Zhao, P.; Tian, H.; Qin, C.; Nie, G. Energy-saving offloading by jointly allocating radio and computational resources for mobile edge computing. *IEEE Access* **2017**, 11255–11268. [CrossRef]
28. Zeng, M.; Fodor, V. Energy minimization for delay constrained mobile edge computing with orthogonal and non-orthogonal multiple access. *Ad Hoc Netw.* **2020**, *98*, 102060. [CrossRef]
29. Meskar, E.; Todd, T.D.; Zhao, D.; Karakostas, G. Energy aware offloading for competing users on a shared communication channel. *IEEE Trans. Mobile Comput.* **2017**, *1*, 87–96. [CrossRef]
30. Wang, S.; Chen, M.; Liu, X.; Yin, C.; Cui, S.; Poor, H.V. A machine learning approach for task and resource allocation in mobile-edge computing-based networks. *IEEE Internet Things* **2021**, *3*, 1358–1372. [CrossRef]
31. Sun, Y.; Zhou, S.; Xu, J. Emm: Energy-aware mobility management for mobile edge computing in ultra dense networks. *IEEE Trans. J. Sel. Areas Commun.* **2017**, *11*, 2637–2646. [CrossRef]
32. Qian, L.P.; Shi, B.; Wu, Y.; Sun, B.; Tsang, D.H.K. NOMA enabled mobile edge com puting for internet of things via joint communication and compu tation resource allocations. *IEEE Internet Things* **2020**, *7*, 718–733. [CrossRef]
33. Liu, L.; Sun, B.; Wu, Y.; Tsang, D.H.K. Latency optimization for computation offloading with hybrid noma-oma transmission. *IEEE Internet Things* **2021**, *8*, 6677–6691. [CrossRef]
34. Chen, G.; Zhao, L.; Li, X.; Zhao, F.; Zeng, X. Optimal Resource Allocation for Multimedia Applications Offloading in Mobile Edge Computing. *IEEE Open J. Comput. Soc.* **2021**, *2*, 360–369. [CrossRef]
35. Zhong, X.; Wang, X.; Li, L.; Yang, Y.; Qin, Y.; Yang, T.; Zhang, B.; Zhang, W. Cl-admm: A cooperative-learning-based optimization framework for resource management in MEC. *IEEE Internet Things* **2021**, *8*, 8191–8209. [CrossRef]
36. Yang, L.; Guo, S.; Yi, L.; Wang, Q.; Yang, Y. NOSCM: A Novel Offloading Strategy for NOMA-Enabled Hierarchical Small Cell Mobile-Edge Computing. *IEEE Internet Things* **2021**, *10*, 8107–8118. [CrossRef]
37. Zhang, J.; Hu, X.; Ning, Z.; Ngai, E.C.-H.; Zhou, L.; Wei, J.; Cheng, J.; Hu, B. Energy-latency tradeoff for energy-aware offloading in mobile edge computing networks. *IEEE Internet Things* **2018**, *4*, 2633–2645. [CrossRef]
38. Li, X.; Zhao, L.; Yu, K.; Aloqaily, M.; Jararweh, Y. A cooperative resource allocation model for IoT applications in mobile edge computing. *Comput. Commun.* **2021**, *173*, 183–191. [CrossRef]
39. Kabir, M.T.; Masouros C. A scalable energy vs. latency trade-off in fullduplex mobile edge computing systems. *IEEE Trans. Commun.* **2019**, *67*, 5848–5861. [CrossRef]
40. Zhu, A.; Wen, Y. Computing offloading strategy using improved genetic algorithm in mobile edge computing syste. *J. Grid Comput.* **2021**, *19*, 38. [CrossRef]
41. Ebadifard, F.; Babamir, S.M. Federated geo-distributed clouds: Optimizing resource allocation based on request type using autonomous and multi-objective resource sharing model. *Big Data Res.* **2021**, *24*, 100188. [CrossRef]
42. Yang, L.; Cao, J.; Tang, S.; Li, T.; Chan, A.T. A framework for partitioning and execution of data stream applications in mobile cloud computing. *Perform. Eval. Rev.* **2013**, *4*, 23–32. [CrossRef]
43. Pochet, Y.; Wolsey, L.A. *Production Planning by Mixed Integer Programming*; Springer: Berlin, Germany, 2006.
44. Li, X.-G.; Han, S.-F.; Zhao, L.; Gong, C.-Q.; Liu, X.-J. New dandelion algorithm optimizes extreme learning machine for biomedical classification problems. *Comput. Intell. Neurosci.* **2017**, *2017*, 4523754. [CrossRef]
45. Lin, N.; Tang, J.; Li, X.; Zhao, L. A novel improved bat algorithm in uav path planning, Computers. *Comput. Mater. Contin.* **2019**, *61*, 323–344. [CrossRef]

46. Yu, Y.; Bu, X.; Yang, K.; Wu, Z.; Han, Z. Green large-scale fog computing resource allocation using joint benders decomposition, dinkelbach algorithm, admm, and branch-and-bound. *Comput. IEEE Internet Things* **2019**, *6*, 4106–4117. [CrossRef]
47. Boyd, S.; Vandenberghe, L. *Convex Optimization*; Computational Cambridge University Press: Cambridge, UK, 2004.
48. Zhu, A.; Wen, Y. Efficient caching strategy in wireless networks with mobile edge computing. *Comput. Peer-to-Peer Netw. Appl.* **2020**, *13*, 1756–1766.
49. Yu, K.; Tan, L.; Lin, L.; Cheng, X.; Yi, Z.; Sato, T. Deep-learning-empowered breast cancer auxiliary diagnosis for 5gb remote e-health. *Comput. IEEE Wirel. Commun.* **2021**, *3*, 54–61. [CrossRef]
50. Xu, Y.; Gu, B.; Hu, R.Q.; Li, D.; Zhang, H. Joint computation offloading and radio resource allocation in MEC-based wireless-powered backscatter communication networks. *Comput. IEEE T. Veh. Technol.* **2021**, *6*, 6200–6205. [CrossRef]
51. Zhao, L.; Zhao, W.; Hawbani, A.; Al-Dubai, A.Y.; Min, G.; Zomaya, A.Y.; Gong, C. Novel online sequential learning-based adaptive routing for edge software-defined vehicular networks. *Comput. IEEE Trans. Wirel. Commun.* **2021**, *5*, 2991–3004. [CrossRef]
52. Yu, K.; Guo, Z.; Shen, Y.; Wang, W.; Lin, J.C.-W.; Sato, T. Secure artificial intelligence of things for implicit group recommendations. *Comput. IEEE Trans. Wirel. Commun.* **2022**, *4*, 2698–2707. [CrossRef]