

## Article

# A Lightweight Secure Adaptive Approach for Internet-of-Medical-Things Healthcare Applications in Edge-Cloud-Based Networks

Abdullah Lakhan <sup>1,2</sup>, Ali Hassan Sodhro <sup>3</sup>, Arnab Majumdar <sup>4</sup>, Pattaraporn Khuwuthyakorn <sup>5</sup> and Orawit Thinnukool <sup>5,\*</sup>

- <sup>1</sup> Department of Computer Science, Dawood University of Engineering and Technology, Karachi 74800, Pakistan; [abdullah.lakhan@duet.edu.pk](mailto:abdullah.lakhan@duet.edu.pk)
- <sup>2</sup> College of Computer Science and Artificial Intelligence, Wenzhou University, Wenzhou 325035, China
- <sup>3</sup> Department of Computer Science, Kristianstad University, SE-291 88 Kristianstad, Sweden; [ali.hassan\\_sodhro@hkr.se](mailto:ali.hassan_sodhro@hkr.se)
- <sup>4</sup> Department of Civil and Environmental Engineering, Imperial College London, London SW7 2AZ, UK; [a.majumdar@imperial.ac.uk](mailto:a.majumdar@imperial.ac.uk)
- <sup>5</sup> College of Arts, Media, and Technology, Chiang Mai University, Chiang Mai 50200, Thailand; [pattaraporn.khuwuth@cmu.ac.th](mailto:pattaraporn.khuwuth@cmu.ac.th)
- \* Correspondence: [orawit.t@cmu.ac.th](mailto:orawit.t@cmu.ac.th)

**Abstract:** Mobile-cloud-based healthcare applications are increasingly growing in practice. For instance, healthcare, transport, and shopping applications are designed on the basis of the mobile cloud. For executing mobile-cloud applications, offloading and scheduling are fundamental mechanisms. However, mobile healthcare workflow applications with these methods are widely ignored, demanding applications in various aspects for healthcare monitoring, live healthcare service, and biomedical firms. However, these offloading and scheduling schemes do not consider the workflow applications' execution in their models. This paper develops a lightweight secure efficient offloading scheduling (LSEOS) metaheuristic model. LSEOS consists of light weight, and secure offloading and scheduling methods whose execution offloading delay is less than that of existing methods. The objective of LSEOS is to run workflow applications on other nodes and minimize the delay and security risk in the system. The metaheuristic LSEOS consists of the following components: adaptive deadlines, sorting, and scheduling with neighborhood search schemes. Compared to current strategies for delay and security validation in a model, computational results revealed that the LSEOS outperformed all available offloading and scheduling methods for process applications by 10% security ratio and by 29% regarding delays.

**Keywords:** neighborhood search; secure offloading; dynamic approaches; workflow healthcare applications; LSEOS; healthcare; scheduling



**Citation:** Lakhan, A.; Sodhro, A.H.; Majumdar, A.; Khuwuthyakorn, P.; Thinnukool, O. A Lightweight Secure Adaptive Approach for Internet-of-Medical-Things Healthcare Applications in Edge-Cloud-Based Networks. *Sensors* **2022**, *22*, 2379. <https://doi.org/10.3390/s22062379>

Academic Editor: Mehmet Rasit Yuce

Received: 26 January 2022

Accepted: 18 March 2022

Published: 19 March 2022

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Copyright:** © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

The development of digital healthcare using the Internet of Medical Things (IoMT) expands day by day as the number of technologies in practice increases [1]. The IoMT is an architectural infrastructure consisting of user, edge, and cloud layers. The user layer consists of sensor devices connected to devices such as mobile devices and smart watches. The edge layer is cloud resources implemented at the edge network, with less communication delay. The cloud layer consists of rich resources and exists multiple hops away from user devices. Many healthcare applications are designed on the basis of service-oriented architecture (SoA) and run in these three layers. These medical applications connect to various biosensors, and send their information to cloud and edge servers for additional evaluation. Heartbeat ECG, EEG, blood pressure, and oxygen levels in the human body, for example, can be monitored by biosensors, and application information is

sent to nearby hospital servers for analysis [2]. To fulfil the healthcare monitoring purpose in the system, the IoT network incorporates diverse elements such as healthcare sensors, wearable networks, and mobile cloud services [3]. The IoT network offloads and shares data of healthcare applications to the different layers, such as network and computing layers, for processing based on their given constraints. However, there are many risks due to other layers in the network [4]. Natural hazards are data security, delays, workload failure due to attacks, and the unavailability of data in the system. Therefore, security-enabled data offloading and processing are important challenges in IoT networks for healthcare applications [5].

Many IoT network-enabled algorithms were suggested for healthcare applications to protect data and store them cryptographically [6]. These static heuristics, such as heterogeneous earliest finish time, genetic algorithm, simulated annealing, and particle swarm optimization, are widely implemented to run healthcare applications in polynomial time [7]. Local and global searches (e.g., simulated annealing and tabu search) are fundamental guided candidate solutions to the aforementioned algorithms to achieve optimal results of healthcare applications in heterogeneous fog cloud networks [8]. There are two types of edge cloud networks widely implemented in the literature, namely, homogeneous and heterogeneous nodes. On the basis of nodes, different studies suggested static and dynamic approaches based on heuristic and guided random search for combinatorial convex optimization research problems of healthcare applications in both heterogeneous and homogeneous environments. Security algorithms implemented inside these heuristics include SHA-256, MD5, CRC32 based on AES and RSA keys in the heterogeneous fog cloud nodes for healthcare applications [9–12].

However, these algorithms present many research challenges when they are implemented inside distributed heterogeneous fog cloud networks for delay-optimal and adaptive healthcare applications in the network. (i) Existing scheduling approaches only focused on resource availability, energy, and security mechanisms in healthcare applications' heterogeneous fog cloud nodes. Static and dynamic scalability can be managed with these methods. However, these methods consume much more resources and energy when they integrate the security mechanism into an IoT network for healthcare applications. (ii) All existing IoT networks have approaches to execute datasets, including healthcare workloads, and they widely ignore workflow healthcare applications in a heterogeneous network. Therefore, IoT network approaches for workflow applications need to be designed in heterogeneous and homogeneous environments for execution. (iii) All existing security approaches are static and consume much more resources in a cryptoprocess on offloaded data from applications. Therefore, a lightweight and adaptive security approach must be designed for healthcare applications in the network.

This study develops the LSEOS metaheuristic, which contains several phases: adaptive sequencing and internal deadlines, adaptive two-way security validation, adaptive task scheduling, and neighborhood search. The goal of the research was to reduce application delays, such as mobile, edge, and cloud execution times. The study examines workflow applications for a variety of jobs, including mobile, edge, and cloud workloads. At the design and development of apps, the tasks are annotated. This research takes into account heterogeneous computing nodes, namely, mobile, edge, and cloud computing.

The manuscript is organized as follows. Section 2 discusses existing IoT network efforts for healthcare applications in homogeneous and heterogeneous fog cloud environments. Section 3 describes the study's proposed architecture and problem formulation. Section 4 shows the flow of proposed heuristics and their steps for the problem solution. Section 5 shows how the proposed work was implemented, and optimal results were obtained compared to existing studies. Section 6 shows the contribution and achievements of the proposed lightweight scheme, and a future road map of the current research.

## 2. Related Work

This part discusses the literature approaches (e.g., static and dynamic) of IoT networks for healthcare applications. Fundamental methods are offloading and scheduling to monitor and schedule healthcare tasks in a system. Metrics and constraints of existing studies are represented in Table 1.

**Table 1.** Existing Internet of Medical Things approaches in IoT networks.

Studies	Approaches	Workload	Fog-Cloud Environment
[1,2]	Genetic Algorithm-SHA-256	Coarse-Grained	Homogeneous
[3,4]	HEFT-PSO-MD5-256	Fine-Grained	Homogeneous
[5,6]	Ant-Colony-CRC-256	Coarse-Grained	Homogeneous
[7,8]	Simulated-Brute-Force-256	Coarse-Grained	Heterogeneous
[9,10]	Genetic Algorithm-DES-256	Coarse-Grained	Heterogeneous
[11,12]	Genetic Algorithm-3DES-256	Coarse-Grained	Heterogeneous
[13,14]	Genetic Algorithm-SHA-256	Coarse-Grained	Heterogeneous
[15,16]	Genetic Algorithm-SHA-256	Coarse-Grained	Heterogeneous
[17,18]	Genetic Algorithm-SHA-256	Coarse-Grained	Homogeneous
[19,20]	Genetic Algorithm-SHA-256	Coarse-Grained	Heterogeneous
[21,22]	LSTM and WMC	Coarse-Grained	Homogeneous
[23,24]	WFMS	Coarse-Grained	Heterogeneous
[25–30]	KNN-DES-256	Coarse-Grained	Heterogeneous

In [1,2], the authors investigated the task offloading and scheduling problem on the basis of a polynomial. These studies suggested genetic-algorithm-based solutions with a secure hashing algorithm (256 bits) for healthcare applications in IoT technologies. These studies obtained different objectives such as response time, tardiness, and network delay. An integrated particle swarm optimization-enabled scheme was presented by [3,4,15,17], where a message digest (MD5) scheme was integrated with a particle swarm optimization (PSO) algorithm to secure healthcare tasks in the IoT network. The goal was to minimize security risks in the network, and these studies considered fine-grained tasks and homogeneous networks in their models. The authors in [5,6] suggested a static improved algorithm based on an ant-colony metaheuristic, and integrated a cryptography algorithm to protect and share secure data between nodes. The considered computing nodes were homogeneous in the IoT network, where these nodes are placed at the edge of the user network.

In [7–16], the authors suggested local and global searching (simulated annealing and genetic algorithm)-enabled dynamic approaches to solve the offloading and scheduling problem in IoT networks. The main goal was to reduce local and global search times for scheduling on heterogeneous fog and cloud nodes in the system, establish a secure environment among connected nodes, and minimize attack risk in the IoT network. These dynamic approaches can identify anomalies and resource performance of applications, but there was still uncertainty of heterogeneous nodes in terms of scalability [17–20].

Machine-learning-enabled convolutional neural network (CNN), k-nearest neighbors (KNN) and support vector machines (SVM) are machine-learning algorithms that support security in adaptive and learning ways. On the basis of these methods, a security mechanism was introduced in IoT networks for the healthcare applications in different works. These studies implemented closely related existing schemes in the simulation, for instance, delay optimal long short-term memory (LSTM) [21], workflow metaheuristic system (WFMS) [22], and workflow metaheuristic cloud (WMC) [23]. These studies are closely related to our work to execute workflow applications on heterogeneous nodes in

cloud computing. In [24–30], the authors suggested dynamic approaches to deal with runtime anomalies and security risks in IoT networks. However, these methods consumed too many resources and had much delay during encryption and decryption in resource allocation.

Many studies [31–35] mobility-enabled fog and edge cloud networks devised for healthcare applications to offer mobility-aware services to mobile patients. The goal was to minimize location offloading risk and unavailability of services in the network. In these works, models offered online remote services from different locations and supported mobility-aware services to patients.

To the best of our knowledge, lightweight secure offloading and scheduling for workflow applications has not been studied yet. The novelty of secure offloading is that the proposed method is lightweight, and consumes fewer resources and time to process security mechanisms and offloading in a mobile edge cloud network. Existing security offloading techniques consume much more resources and time when processing IoT healthcare applications in a network. Therefore, the dynamic and secure offloading scheme is lightweight and robust, and meets the requirements of IoT healthcare applications in the network. Generally, dynamic scheduling methods are those in which tasks are prioritized at runtime in the system.

### 3. Proposed LSEOS Metaheuristic and Architecture

As illustrated in Figure 1, a lightweight and delayed optimal secure Internet of Medical Things (IoMT) network was devised for workflow healthcare applications. The goal was to provide optimal offloading and scheduling in a network that are both secure and delay-optimal. The suggested architecture comprises layers for IoT (i.e., IoMT) workflow applications, management, and resources. At the design time of applications, the program was the workflow and consisted of three processor tasks in the system, namely, local, fog, and cloud tasks. Local tasks must be completed on mobile devices with the least delay and amount of time while sharing data with the system’s edge node tasks.

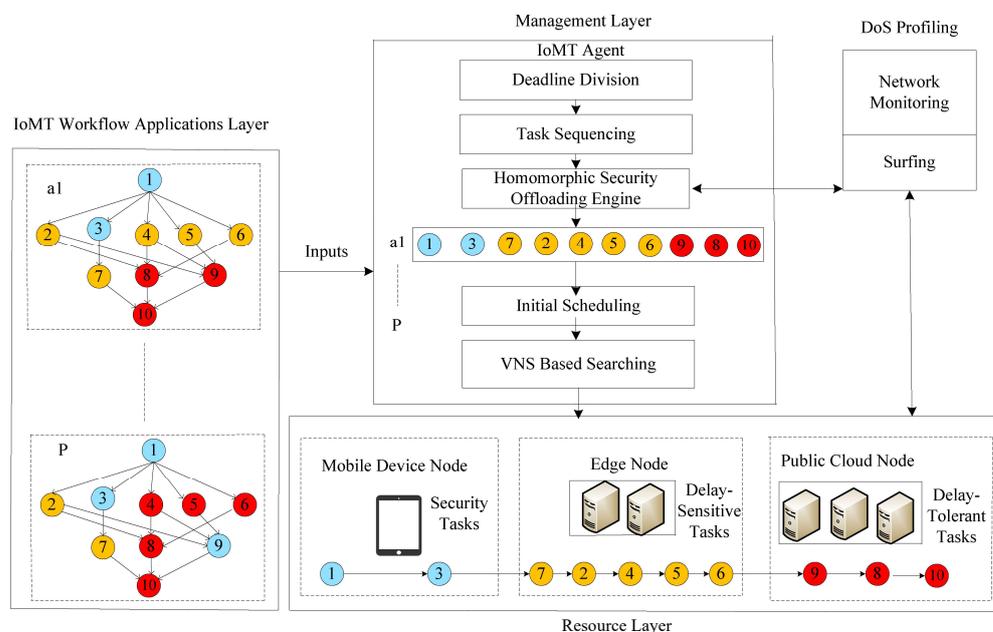


Figure 1. LSEOS metaheuristic and architecture.

Edge tasks are simultaneously scheduled at available edge nodes for further processing and data collection from local tasks before being executed. The cloud tasks merely store the application’s data-intensive results in the cloud without compromising the system’s application performance. The IoMT network’s management layer comprises various dy-

namic offloading and scheduling methods. The IoMT agent, which is more adaptive and handles all application operations at any time in the system, was devised in this study. The IoMT agent is the primary handler and combines several approaches: deadline division, topological order sequencing, a fully homomorphic mechanism, preliminary task assignment (initial scheduling), and variable neighborhood adaptive resource seeking for tasks. Denial of service (DoS)-aware techniques were developed on the basis of network surfing and system monitoring without compromising the system's application performance. The problem's symbolic notations and descriptions are defined in Table 2.

**Table 2.** Problem notations.

Symbol	Purpose
$P$	Number of workflow applications
$A$	Particular workflow application
$m_i$	Mobile tasks of workflow application $a$ (local tasks)
$i$	Specific task of application $a$
$e_i$	Edge tasks of workflow $a$ (delay-sensitive tasks)
$c_i$	Cloud computing tasks of workflow application $a$ (delay-tolerant tasks)
$w_a$	Workload of workflow application $a$
$deadline_a$	Deadline of workflow $a$
$m$	Mobile node
$e$	Edge/fog node
$c$	Cloud node
$M_e^{mi}$	Execution time of a task at mobile device
$E_e^{ei}$	Execution time of a task at edge computing
$C_e^{ci}$	Execution time of a task at cloud computing
$\zeta_m$	Mobile computing node
$\epsilon_m$	Resource of mobile node
$\zeta_k$	Edge node speed
$\epsilon_k$	Resource of edge node
$\zeta_c$	Cloud computing node
$\epsilon_c$	Resource of cloud computing node
$me$	Communication time between mobile and edge nodes
$ec$	Communication between edge computing and cloud computing
$x_{mi=1}$	Assignment of tasks to mobile device
$x_{ei=2}$	Assignment of tasks to edge computing
$x_{ci=3}$	Assignment of tasks to cloud computing

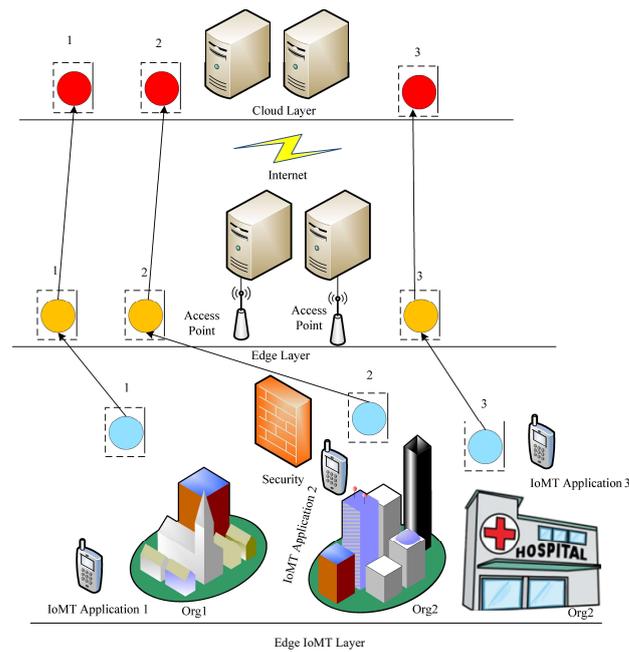
### 3.1. Problem Formulation and System Model

The problem was formulated as follows. The execution scenario of workflow applications in an IoMT network was first discussed as shown in Figure 2.

Real practical scenarios of healthcare hospitals as different organizations (Org) were considered. Many healthcare sensors were connected with varying healthcare organizations, and their data were offloaded for processing via access points at edge computing. Local tasks were executed in the IoT (mobile devices), and edge tasks were offloaded to the edge computing layer for execution. At the same time, delay-tolerant tasks were offloaded to the cloud via the Internet, as shown in Figure 2.

The mathematical formulation of the study was as follows.  $P$  number of workflow applications were represented by a directed acyclic graph. The workflow application had starting tasks  $i$  and  $j$ , and there was communication between them. Each workflow application had workload  $w_a$  and deadline  $deadline_a$ .

Three different computing nodes were considered: mobile device  $m$ , edge server  $e$ , and cloud computing  $c$ . Each node had a different processing speed  $\zeta_m, \zeta_e, \zeta_c$ , and resources  $\epsilon_m, \epsilon_e, \epsilon_c$  in the system.



**Figure 2.** IoMT system model.

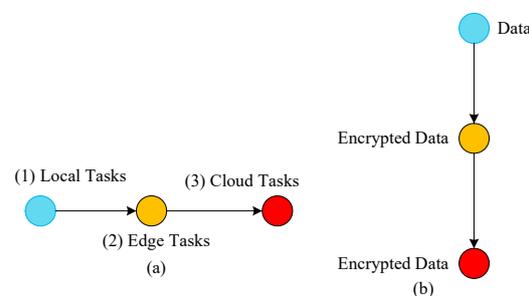
### 3.2. Task Annotation at Design Time

The application divides the workload into three task types: mobile, edge, and cloud tasks. Mobile tasks must be locally executed on mobile devices. These tasks are locally encrypted and decrypted with the proposed secure algorithm. Then, encrypted data are offloaded to edge tasks for execution. After executing tasks, the data are offloaded to cloud computing to complete cloud tasks in the system.

### 3.3. Workflow Application Characterization

The considered application contained three different task types: mobile (represented by blue nodes), edge (yellow nodes), and cloud (red nodes) tasks, as shown in Figure 3.

Figure 3a denotes the sequence execution of tasks from local to cloud tasks with different requirements. Due to data security, local tasks encrypt and decrypt locally at the local machine, and then offload their cipher data to edge tasks for further execution. Cipher data are never interpreted at edge nodes, and computation is applied on the cipher data to complete their process. Edge nodes, on the other hand, send their executed cipher data to the cloud node for further storage as shown in Figure 3b.



**Figure 3.** Workflow applications on distributed mobile edge cloud networks in Proposed Architecture.

### 3.4. Mobile Computing Assignment

In this section, we determine the execution time of  $mi$  tasks at local devices, which locally had encryption and decryption time at mobile devices. Local execution time is determined for local mobile tasks in the following way.

$$M_e^{mi} = \sum_{mi=1}^a \frac{Enc}{\zeta_m} + \frac{Dec}{\zeta_m} \times x_{mi=1} \quad (1)$$

Equation (1) determines the encryption and decryption time of local tasks at mobile device. For homomorphic encryption, cryptographic modeling was suggested on the basis of the El Gamal scheme. The homomorphic mechanism consists of three main elements: (1) key generation, (2) encryption time, and (3) a decryption-time-based asymmetric public key. Encryption and decryption based on an asymmetric public key are locally determined in the following way.

$$Enc = Pk \leftarrow data_i \leftarrow SHA - 256. \quad (2)$$

Equation (2) determines the encryption time (PK(public key) in the system.

$$Dec = PV \leftarrow Enc \leftarrow data_i. \quad (3)$$

Equation (3) determines the decryption time with the private key (PV) in the system.

### 3.5. Edge Computing Assignment

Data are offloaded from the mobile device to the edge node in the form of cipher text. Therefore, the execution time on the edge nodes consists of communication and processing times between mobile devices, and the edge is determined in the following way.

$$E_e^{ei} = \sum_{ei=1}^a \frac{Enc}{\zeta_e} + \frac{data_i \leftarrow Enc}{bw_{upload}} \leftrightarrow \frac{data_i \leftarrow Enc}{bw_{download}} \times x_{ei=2}. \quad (4)$$

Equation (4) determines the execution time of encrypted tasks, and the communication between mobile device and edge with both uploading and downloading data.

### 3.6. Cloud Computing Assignment

Cloud computing only stores generated data from the edge nodes. Therefore, execution on encrypted executed data for uploading and downloading is determined in the following way.

$$C_e^{ci} = \sum_{ci=1}^a \frac{Enc}{\zeta_c} + \frac{data_i \leftarrow Enc}{bw_{upload}} \leftrightarrow \frac{data_i \leftarrow Enc}{bw_{download}} \times x_{ci=3} \quad (5)$$

Equation (5) determines the execution time of encrypted tasks, and communication between cloud and edge with both uploading and downloading data. The total execution and communication times of all tasks on all nodes are determined in the following way.

$$Delay = \sum_{a=1}^P \sum_{mi=1}^a \sum_{ei=1}^A \sum_{ci=1}^A M_e^{mi} + E_e^{ei} + C_e^{ci}, \quad \forall a = 1, \dots, P. \quad (6)$$

Equation (6) calculates the total delay time of workflow application  $a$  at different computing nodes. The problem formulation based on linear programming was designed in the following way.

$$\min Delay, \quad a = 1, \dots, P. \quad (7)$$

Equation (7) is the combinatorial convex objective of the considered problem in the system.

$$\sum_{a=1}^{m_i, e_i, c_i} data_i \leq \epsilon_{m, e, c}. \quad (8)$$

Equation (8) ensures that, before execution on the application, there are enough resources in all computing nodes for the processing.

$$\sum_{a=1}^P data_i \leq deadline_a, \quad a = 1, \dots, P. \quad (9)$$

All applications must be executed before their deadlines as defined in Equation (9).

#### 4. Proposed Security Efficient Offloading and Task-Scheduling (LSEOS) Metaheuristic Approach

The combinatorial convex optimization problem for heterogeneous tasks and parallel computing nodes is always challenging. This study also considers different workflow tasks such as mobile, delay-sensitive, and delay-tolerant, and heterogeneous computing nodes such as mobile devices, edge nodes, and cloud computing in the system. The goal was to execute workflow applications on different nodes in order to minimize the total delay of applications. Existing offloading schemes [1,4,7] only focused on coarse- and fine-grained applications. Therefore, there are no particular architectures and schemes for workflow applications. This study devised a lightweight security-enabled efficient offloading and scheduling (LSEOS) algorithm framework that consists of different schemes to minimize the delay of applications in the architecture. The proposed SEOS, as shown in Algorithm 1, comprises different schemes, and was determined as follows.

---

##### Algorithm 1: LSEOS metaheuristic.

---

**Input** :  $P = \{a = 1, \dots, aP\}, m, e, c$

**Output**: min Delay;

1 **begin**

2     **foreach** ( $a$  in  $P$ ) **do**

3         Call-Internal-Task-Sequence Scheme;

4         Call Two-Way Homomorphic Scheme;

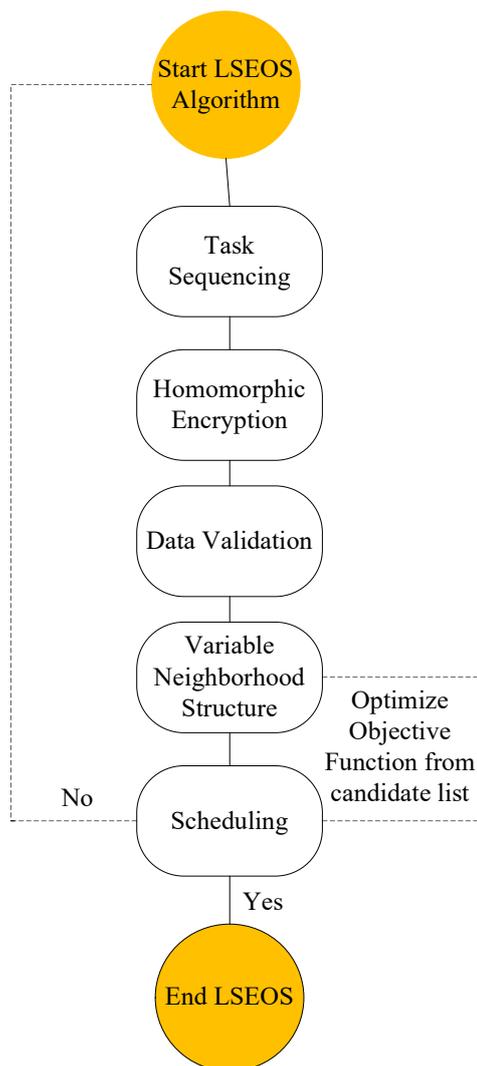
5         Call Two-Way Data Validation Scheme;

6         Call Adaptive Variable Neighborhood Structure Scheme;

7         Call Adaptive Scheduling Scheme;

---

Figure 4 shows the LSEOS metaheuristic process from the initial to the end component for the considered problem. It starts from input and ordering all tasks on the basis of their quality-of-service (QoS) requirements in the system. The primary goal of the sequence is to order all tasks on the basis of their priorities such as deadlines and delays in the system. Once the tasks are sorted, there is two-way homomorphic encryption. This encryption mechanism encrypts and decrypts data at local devices. Fog and cloud nodes process these tasks on the basis of their ciphertext instead of plaintext in the system. Two-way validation ensures the validation of data between nodes. The candidate solution is the searching mechanism inside LSEOS that finds random optimal solutions and replaces the existing scheduling solution when its status is "Yes". If tasks are failed, it generates a "No" status and reprocesses all tasks from start in the system, as shown in Figure 4.



**Figure 4.** LSEOS metaheuristic algorithm for the considered problem.

#### 4.1. Adaptive Task Sequencing Rule Scheme

There are many sorting techniques that are widely exploited in the workflow applications, for instance, deadline-enabled sorting, earliest deadline finish time, earliest finish time, and smallest workload first. However, these techniques only sort tasks on a single group of same applications. However, workflow application tasks are divided into three sets: local, delay-sensitive, and delay-tolerant tasks. Therefore, these methods cannot be directly applied on different sets of tasks, and internal adaptive task sequencing rules are suggested in which all tasks are sorted according to sets.

Algorithm 2 initially assigns a deadline to tasks at different levels, such as mobile, delay-sensitive, and cloud tasks, on the basis of their execution time. All tasks are sorted on the basis of their deadlines. Assigned deadline and sorting are adaptive rules that assign the deadline and sorting to workflow applications at the runtime of submission in the system. Algorithm 2 determines the deadline and sequencing rules in the following way. Deadline:  $a = \{i = 1 \leftarrow 30 s, i = 3 \leftarrow 40 s, i = 2 \leftarrow 30 s, i = 4 \leftarrow 50 s, i = 5 \leftarrow 60 s, i = 6 \leftarrow 30 s, i = 9, i = 8, i = 10 \leftarrow 89 s\}$ . After deadline assignment, tasks are sorted as follows:  $a = \{i = 1, i = 3, i = 2, i = 4, i = 5, i = 6, i = 9, i = 8, i = 10\} \forall a = 1, \dots, A$ .

**Algorithm 2:** Adaptive task sequencing rule scheme.

---

```

Input :  $P = \{a = 1, \dots, aP\}$ 
1 begin
2   foreach ( $a$  in  $P$ ) do
3     Determine execution time of task  $i$  based on Equation (1);
4     Assign deadline of a task on the basis of execution time.
5      $d_{mi} = x_{i,m} \leftarrow deadline_a \leftarrow A$ ;
6     All tasks are ordered on the basis of their assigned deadline;
7     Determine the execution time of task  $i$  on the basis of Equation (4);
8     Assign deadline of a task on the basis of their execution time.
9      $d_{ei} = x_{i,e} \leftarrow deadline_a \leftarrow A$ ;
10    All tasks are ordered on the basis of their assigned deadline;
11    Determine the execution time of task  $i$  on the basis of Equation (5);
12    Assign deadline of a task on the basis of their execution time.
13     $d_{ci} = x_{i,c} \leftarrow deadline_a \leftarrow P$ ;
14    All tasks are ordered on the basis of their assigned deadline;

```

---

## 4.2. Two-Way Secure Offloading Scheme

Algorithm 3 determines the two-way homomorphic security mechanisms where denials of service are monitored via network profiler scheme before offloading any data from any node. Surfing is a mechanism that generates the report of intrusion nodes in the system. Algorithm 3 determines the stability of nodes before offloading any data to any node. Data for the encryption are computed on the basis of a 256-bit public key, and two random long integers always generate random keys on given task data in the algorithm. The network profiler always checks the status of security in the algorithm. The data can be decrypted on the basis of a private key at the mobile devices, and results are accessed from the cloud node.

**Algorithm 3:** Lightweight two-way homomorphic security scheme.

---

```

Input :  $P = \{a = 1, \dots, aP\}, m$ 
Output: min Delay;
1 begin
2   Initially, Key – Gen (256-bits) numbers;
3    $p, q$  are random long integers;
4   Network monitoring NW;
5   Dos profiling DP;
6   Surfing SR;
7   foreach ( $a$  in  $P$ ) do
8      $a = \{i = 1, i = 3, i = 2, i = 4, i = 5, i = 6, i = 9, i = 8, i = 10 \in A\}$ ;
9     Compute encryption on the basis of a public key;
10     $Encryption[data_i, a \in P] = (data_i, m, Key - Gen, p, q, Public - Key) \leftarrow a$ 
11    Equation (2);
12    if ( $NW \leftarrow DP || SR == valid$ ) then
13       $\leftarrow$  Offload  $\leftarrow Encryption[data_i, m]$ ;
14    else
15       $\leftarrow$  Cannot offload data to the edge node;
16   $a = \{i = 1, i = 3, i = 2, i = 4, i = 5, i = 6, i = 9, i = 8, i = 10 \in A\}$ ;
17  Compute encryption on the basis of a public key;
18   $Decryption[data_i, m, Key - Gen, p, q, Private - Key] \leftarrow a$ 
19  Equation (3) =  $Encryption[d_i, a \in P]$ ;
20  if ( $NW \leftarrow DP || SR == valid$ ) then
21     $\leftarrow$  Offload  $\leftarrow Encryption[data_i, m]$ ;
22  else
23     $\leftarrow$  Cannot download data from cloud node;

```

---

#### 4.3. Adaptive Task Scheduling and Neighborhood Structure

The two ways of offloading are from mobile devices to edge node and from edge node to cloud computing. Adaptive task scheduling is a mechanism where execution time is divided among mobile devices, edge nodes, and cloud computing. There are many existing algorithms that can schedule different nodes in the literature, for instance, heterogeneous earliest finish time algorithm, genetic algorithm, PSO, and ant colony. With these algorithms, heterogeneous jobs can be run on heterogeneous cloud nodes. However, these algorithms cannot work with encryption at the edge and cloud nodes to process workflow applications. Workflow applications have different requirements, for instance, encryption can be performed on one node, and other nodes must compute the encrypted data and not the plaintext. A novel task-scheduling and variable neighbour-searching scheme, Algorithm 4, was devised.

---

#### Algorithm 4: Adaptive task-scheduling and variable neighbor-searching scheme.

---

```

Input :  $m, e, c, Encryption[data_i, m]$ 
1 begin
2   Sort all resources by their available resources;
3   based on Equation (8);
4   Schedule all tasks on the basis of Equation (9);
5   Compute  $m, e, c, Encryption[data_i, m]$ ;
6   if ( $Encryption[data_i, m] \leq d_i$ ) then
7     Apply computation on ciphertext on the basis of Equation (4);
8      $result = Encryption[data_i, m] \in a \in A$ ;
9     Optimize Delay  $\leftarrow result$  on the basis of Equation (7);
10    After 100 s, resources are rearranged again;
11    Find new solution  $data'$  on the basis of Equation (6);
12    if ( $data' \leq data$ ) then
13      replace  $data \leftarrow data'$ ;
14      Current solution is  $data'$ ;
15    else
16      Running solution  $data$  is optimal;
17    End of Inner
18 End of Main;

```

---

#### 5. Performance Evaluation

In this section, existing performance workflow schemes [15,17,19,20] are compared with proposed scheme LSEOS on the basis of the performance of workflow applications. The IFogsim framework was used, where proposed and existing approaches were efficiently implemented and their performance in the architecture was evaluated. Table 3 shows the parameter settings of the proposed architecture, which was implemented on the basis of IFogsim for the experiments.

**Table 3.** Lfogsim-based experimental parameters.

Simulation Parameters	Values
Setup	IFogsim
Workflow	Healthcare config file
Evaluation Methods	Statistical median
Mobile Node	HTC G17 and Samsung 1997
Edge Node	Intel 5 laptop, AndroidX86
Public Node	AndroidX86 Amazon t2.medium
<i>me</i>	10–30 ms
<i>ec</i>	100 ms
Performance criteria	Security and delay, deadline, and data validation based on Equations (6) and (9).

### 5.1. Parameter Settings of Simulation Environment of the Considered Problem

In this part, the study showed the simulations parameters of the considered problem and showed how to conduct the experiments based on the proposed and baseline approaches.

To evaluate the performance of IoMT workflow tasks, we designed different flows of different types of workflow tasks, namely, a set of security, delay-sensitive, and delay-tolerant tasks at design time. Security must be locally executed with private encrypted and decrypted keys, delay-sensitive tasks are offloaded to the edge node for execution, and delay-tolerant tasks should be executed onto the remote cloud. All types are represented by different nodes. Blue nodes are security tasks, light yellow nodes are delay-sensitive tasks, red circles show delay-tolerant tasks, and other types were randomly designed.

Table 4 denotes the workflow applications and their tasks. Each workflow application consisted of three types of tasks, as shown in Table 4: mobile, edge, and cloud tasks; they are processed at different nodes in the system. All tasks were part of a workflow, some had original data, and some shared their data for processing. All tasks were constrained by their predecessor and successor in the system.

**Table 4.** Workflow tasks.

Workflow Application	Mobile Tasks	Edge Tasks	Cloud Tasks	File
a1	100	500	1000	Configuration
a2	50	400	700	Configuration
a3	90	600	800	Configuration

The objective function of the study, that is, delay measured in terms of microseconds (ms) for workload assignment to the distributed nodes. We now compare the results of IoMT workflow tasks with the proposed framework, its components, and existing offloading and scheduling frameworks. Component results are discussed below.

### 5.2. Delay Optimal Result Comparison

Obtained results were based on delay that was calculated in microseconds (ms), as shown in Table 5. Execution delay (ms) and deadlines for all applications are shown in Table 5. LSEOS outperformed all existing methods in terms of delay and deadline, as shown in Figure 5.

**Table 5.** Delay optimal workload assignment.

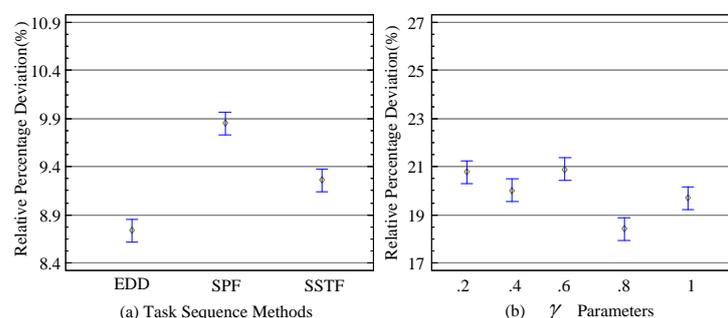
Application	Methods	Delay (ms)	Deadline (ms)
a1	LSEOS	1000	899
a1	LSTM Metaheuristic	1300	899
a1	WMC	1500	899
a1	WFMS	1000	899
a2	LSEOS	700	1000
a2	LSTM Metaheuristic	850	1000
a2	WMC	1200	1000
a2	WFMS	1100	1000
a3	LSEOS	320	500
a3	LSTM Metaheuristic	450	500
a3	WMC	700	500
a3	WFMS	670	500

### 5.3. Workflow Task Deadline Division

The workflow divided deadlines into tasks on the basis of their execution and communication time. This technique helps in how to execute all tasks under their deadlines onto different nodes. In the experiment, we divided the single workflow into task deadlines on the basis of Algorithm 2. All divisions were performed before offloading and scheduling in the system.

### 5.4. Task Sequencing Rules of Workflow Tasks

The tightness of values start from 0.2 not 0 because, in our model we show that 0.2 is the initial deadline and 0 means not scheduled in the distributed mobile edge cloud network. The task sequence component is important before task scheduling onto heterogeneous computing nodes. Different topological task sequencing rules are proposed that consist of EDD, SPD, and SSTF rules. Due to the different characteristics of tasks, the deadline, security, and availability of resources sorting all tasks with one rule is not enough. Therefore, we sorted all tasks into three sequence rules. All tasks were first sorted on the basis of EDD, and then they were sorting-based SPD. Lastly, on the basis of the best availability of resources, we sorted all tasks on the SSTF rule as shown in Figure 5a; EDD worked excellently as compared to others.

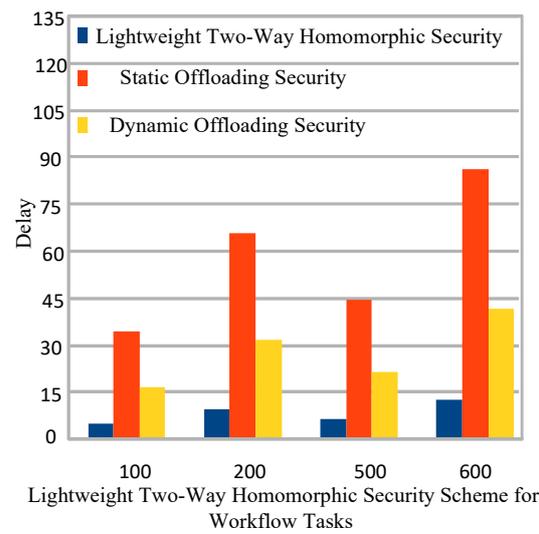
**Figure 5.** Task sequencing Tightness Rules With Different Rules.

### 5.5. Assignment Delay Performance

These studies implemented as the baseline approaches that are closely related existing schemes in the simulation, for instance, delay optimal long short-term memory (LSTM) [21], workflow metaheuristic system (WFMS) [22], and workflow metaheuristic cloud (WMC) [23]. These studies are closely related to our work to execute the workflow applications on the heterogeneous nodes in mobile edge cloud computing.

### 5.6. Lightweight Secure Offloading

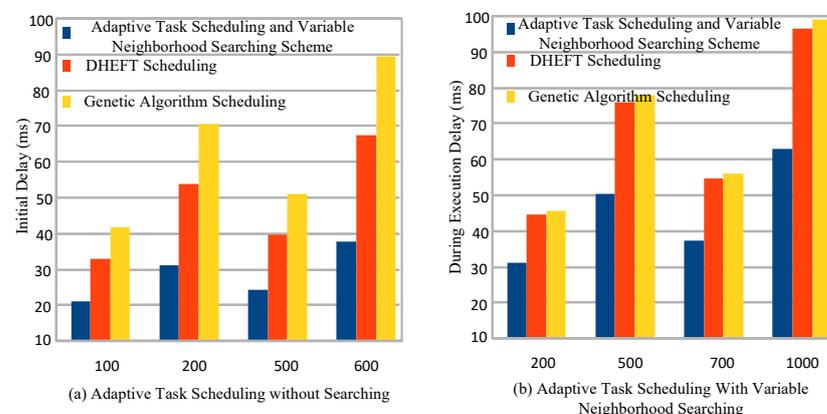
The lightweight mechanism shows the less delay execution during secure offloading between nodes in the study. The study devises the two secure lightweight homomorphic encryption secure scheme which only encrypt and decrypt data at the mobile devices instead of fog node and cloud. The main goal is to minimize the security delay at different level of nodes as existing static [11] and dynamic [1] offloading did in for the workflow applications. Figure 6 shows that, the proposed two ways lightweight homomorphic secure scheme outperformed all existing schemes for the mobile workflow applications in the system.



**Figure 6.** Lightweight Secure Offloading of Workflow Applications.

### 5.7. Adaptive Task Scheduling

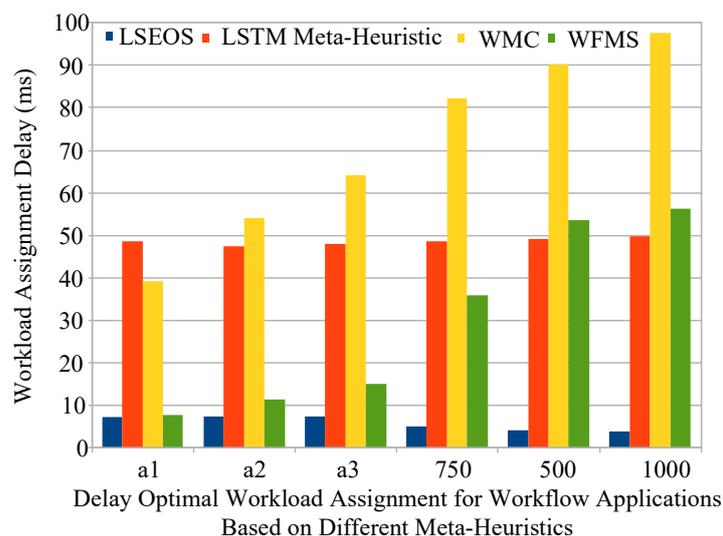
The scheduling schemes DHEFT [1] and genetic algorithm [5] suggested the static and dynamic scheduling algorithms to run the workflow applications in mobile edge cloud network. The study devises the initial scheduling at different nodes gain the lower delay as shown in Figure 7a gain lower delay as compared to existing both DHEFT and genetic scheduling schemes. Figure 7b shows the adaptive delay optimal searching nodes based neighborhood searching enabled scheduling in the distributed mobile fog cloud network. The result shows that, the proposed adaptive searching technique search the delay optimal nodes at the run time of workflow applications in the system.



**Figure 7.** Adaptive task-scheduling and neighborhood-searching scheme for workflow applications.

### 5.8. Delay Optimal Task Assignment

Figure 8 (a) analyzes the delay performance of all offloading and schemes on workflow applications in a heterogeneous mobile cloud environment. Proposed scheme LSEOS outperformed all existing schemes of LSTM, WMC, and WFMS metaheuristics with all workflow applications in terms of delay and workload assignment in mobile fog cloud networks. There are many reasons why these schemes have higher delay as compared to that of the proposed scheme. These studies suggested workflow schemes based on single node either on the mobile device, edge node or cloud computing. Due to the high complexity of workflows and their intermediate task dependency, the workflow application could not be at same place because their applications are so much heavier. For instance, mobile devices cannot run these applications alone at the local device due to resource-constraint issues. However, edge nodes can only run the workflow with minimal delay, as data cannot be stored on edge nodes due to their limited resources in the network. Therefore, cloud computing can store data, but due to the long distance, the cloud has longer execution and communication delay for workflow applications. The LSEOS divided applications into three different parts and achieved less delay due to the division of applications and executing them under their deadline.

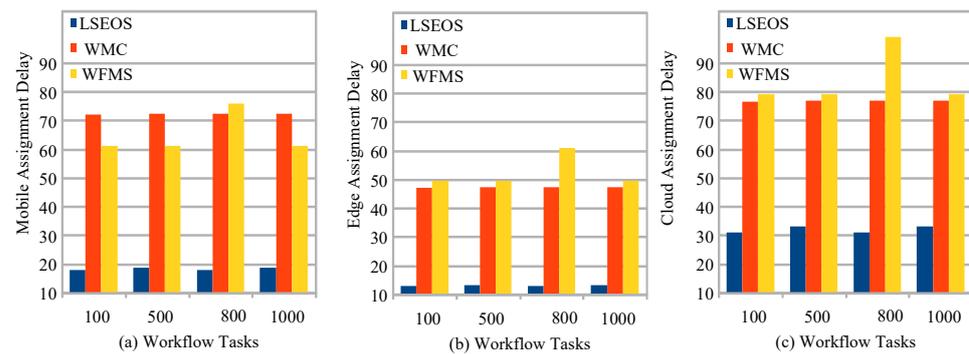


**Figure 8.** Lightweight workflow task assignment delay performance.

### 5.9. Lightweight Secure Adaptive Offloading and Scheduling Approaches

This section outlines the achievements of methods that are lightweight secure adaptive offloading and scheduling approaches in the system.

Figure 9a–c show the lightweight workflow task assignment performance with the security validation and their deadline in the heterogeneous mobile edge cloud. The result shows that, the LSEOS outperformed all existing schemes because of the lightweight security scheme as compared existing secure offloading and scheduling in terms of delay for the healthcare workflow applications. LSEOS only encrypts and decrypts on local devices; therefore, it achieved a high security ratio, and edge and cloud nodes do not need to decrypt them. In this way, encryption and decryption time, and security validation can be widely improved for workflow applications when they are executed on different heterogeneous nodes.



**Figure 9.** Lightweight secure adaptive offloading and scheduling workflow task performance: (a) Mobile Assignment Delay, (b) Edge Assignment Delay, (c) Cloud Assignment Delay.

### 5.10. Findings and Limitations

This paper devised the lightweight, secure offloading and task scheduling (LSEOS) algorithm framework, consisting of different schemes. The objective is to run workflow applications on other nodes and minimize the delay and security risk in the system. An adaptive deadline, sorting, and scheduling with neighborhood search schemes are suggested in the proposed model. This study found the following: (1) a lightweight security method was designed that minimized the security resources and time for the applications; (2) processing delays were minimized during offloading and scheduling in the system; (3) different sequencing rules were devised that satisfied the deadlines, priorities, and quality of service of applications. However, there are a few limitations to the proposed work. The study only supported the workflow application and did not support the coarse- and fine-grained workload in the system. Furthermore, the processing cost and security cost enabled constraints are very important during offloading and scheduling and were not considered in the system.

## 6. Conclusions and Future Work

This paper devised the lightweight secure offloading and task scheduling (LSEOS) algorithm framework, which consists of different schemes. The objective is to run workflow applications on other nodes and minimize the delay and security risk in the system. The adaptive deadline, sorting, and scheduling with neighborhood search schemes are suggested in the proposed model. The simulation results showed that the SEOS outperforms all existing offloading and scheduling methods for the workflow applications compared to current techniques for the delay and security validation in the model. There are many challenges exist in the current manuscript version that did not consider the block to block type security and validation in the proposed architecture. Mobility aspects of users were not investigated in the current architecture; therefore, future work will consider node to node validation and mobility of the users in the proposed architecture.

**Author Contributions:** A.L., A.H.S. and O.T.: conceptualization, investigation, methodology, resources, software, writing—original draft; A.L., A.H.S. and O.T.: investigation, validation, review and editing; P.K. and A.M.: conceptualization, software, writing—review and editing; A.L., A.H.S., P.K. and O.T.: writing—review and editing. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research has received funding support from the NSRF via the Program Management Unit for Human Resources & Institutional Development, Research and Innovation (Grant number B16F640189), and Chiang Mai University Grant number R000029859.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** All the experimental data are generated at the local institution servers. Therefore, it cannot be made publicly available for other researchers.

**Acknowledgments:** This research work was partially supported by Dawood University of Engineering and Technology and Chiang Mai University.

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

## References

1. Huang, B.; Li, Z.; Tang, P.; Wang, S.; Zhao, J.; Hu, H.; Li, W.; Chang, V. Security modeling and efficient computation offloading for service workflow in mobile edge computing. *Future Gener. Comput. Syst.* **2019**, *97*, 755–774. [[CrossRef](#)]
2. Wang, W.; Tornatore, M.; Zhao, Y.; Chen, H.; Li, Y.; Gupta, A.; Zhang, J.; Mukherjee, B. Infrastructure-efficient virtual-machine placement and workload assignment in cooperative edge-cloud computing over backhaul networks. *IEEE Trans. Cloud Comput.* **2021**, *3*, 1–10. [[CrossRef](#)]
3. Liang, J.; Li, K.; Liu, C.; Li, K. Joint offloading and scheduling decisions for DAG applications in mobile edge computing. *Neurocomputing* **2021**, *424*, 160–171. [[CrossRef](#)]
4. Al-Habob, A.A.; Dobre, O.A.; Armada, A.G.; Muhaidat, S. Task scheduling for mobile edge computing using genetic algorithm and conflict graphs. *IEEE Trans. Veh. Technol.* **2020**, *69*, 8805–8819. [[CrossRef](#)]
5. Lakhan, A.; Li, X. Transient fault aware application partitioning computational offloading algorithm in microservices based mobile cloudlet networks. *Computing* **2020**, *102*, 105–139. [[CrossRef](#)]
6. Lakhan, A.; Ali Dootio, M.; Sodhro, A.H.; Pirbhulal, S.; Groenli, T.M.; Khokhar, M.S.; Wang, L. Cost-efficient service selection and execution and blockchain-enabled serverless network for internet of medical things. *Math. Biosci. Eng.* **2021**, *18*, 7344–7362. [[CrossRef](#)]
7. Lakhan, A.; Mohammed, M.A.; Kozlov, S.; Rodrigues, J.J. Mobile-fog-cloud assisted deep reinforcement learning and blockchain-enabled IoMT system for healthcare workflows. *Trans. Emerg. Telecommun. Technol.* **2021**, *18*, e4363. [[CrossRef](#)]
8. Lakhan, A.; Memon, M.S.; Mastoi, Q.-U.-A.; Elhoseny, M.; Mohammed, M.A.; Qabulio, M.; Abdel-Basset, M. Cost-efficient mobility offloading and task scheduling for microservices IoVT applications in container-based fog cloud network. *Clust. Comput.* **2021**, *8*, 1–23. [[CrossRef](#)]
9. Lakhan, A.; Dootio, M.A.; Alqahtani, F.; R Alzahrani, I.; Baothman, F.; Shah, S.Y.; Shah, S.A.; Anjum, N.; Abbasi, Q.H.; Khokhar, M.S.; et al. Hybrid workload enabled and secure healthcare monitoring sensing framework in distributed fog-cloud network. *Electronics* **2021**, *10*, 1974. [[CrossRef](#)]
10. Lakhan, A.; Mohammed, M.A.; Rashid, A.N.; Kadry, S.; Panityakul, T.; Abdulkareem, K.H.; Thinnukool, O. Smart-Contract Aware Ethereum and Client-Fog-Cloud Healthcare System. *Sensors* **2021**, *21*, 4093. [[CrossRef](#)]
11. Han, S.; Xu, X.; Fang, S.; Sun, Y.; Cao, Y.; Tao, X.; Zhang, P. Energy efficient secure computation offloading in NOMA-based mMTC networks for IoT. *IEEE Internet Things J.* **2019**, *6*, 5674–5690. [[CrossRef](#)]
12. Hussain, M.; Wei, L.F.; Lakhan, A.; Wali, S.; Ali, S.; Hussain, A. Energy and performance-efficient task scheduling in heterogeneous virtualized cloud computing. *Sustain. Comput. Inform. Syst.* **2021**, *30*, 100517. [[CrossRef](#)]
13. Lakhan, A.; Khoso, F.H.; Arain, A.A.; Kanwar, K. Serverless based functions aware framework for healthcare application. *Int. J. Emerg. Trends Eng. Res.* **2021**, *9*, 446–450.
14. Khoso, F.H.; Lakhan, A.; Arain, A.A.; Soomro, M.A.; Nizamani, S.Z.; Kanwar, K. A Microservice-Based System for Industrial Internet of Things in Fog-Cloud Assisted Network. *Eng. Technol. Appl. Sci. Res.* **2021**, *11*, 7029–7032. [[CrossRef](#)]
15. Wang, Y.; Zuo, X. An effective cloud workflow scheduling approach combining PSO and idle time slot-aware rules. *IEEE/CAA J. Autom. Sin.* **2021**, *8*, 1079–1094. [[CrossRef](#)]
16. Tang, C.; Hao, M.; Wei, X.; Chen, W. Energy-aware task scheduling in mobile cloud computing. *Distrib. Parallel Databases* **2018**, *36*, 529–553. [[CrossRef](#)]
17. Neforawati, I.; Arnaldy, D. Message Digest 5 (MD-5) Decryption Application using Python-Based Dictionary Attack Technique. In Proceedings of the 2021 4th International Conference of Computer and Informatics Engineering (IC2IE), Depok, Indonesia, 14–15 September 2021; pp. 424–428.
18. Tang, C.; Xiao, S.; Wei, X.; Hao, M.; Chen, W. Energy Efficient and Deadline Satisfied Task Scheduling in Mobile Cloud Computing. In Proceedings of the 2018 IEEE International Conference on Big Data and Smart Computing (BigComp), Shanghai, China, 15–17 January 2018; pp. 198–205.
19. Peng, H.; Wen, W.S.; Tseng, M.L.; Li, L.L. Joint optimization method for task scheduling time and energy consumption in mobile cloud computing environment. *Appl. Soft Comput.* **2019**, *80*, 534–545. [[CrossRef](#)]
20. Zhou, B.; Dastjerdi, A.V.; Calheiros, R.N.; Buyya, R. An online algorithm for task offloading in heterogeneous mobile clouds. *ACM Trans. Internet Technol.* **2018**, *18*, 1–25. [[CrossRef](#)]
21. Arif, M.; Ajesh, F.; Shamsudheen, S.; Shahzad, M. Secure and Energy-Efficient Computational Offloading Using LSTM in Mobile Edge Computing. *Secur. Commun. Netw.* **2022**, *2022*, 4937588. [[CrossRef](#)]
22. Abbas, A.; Raza, A.; Aadil, F.; Maqsood, M. Meta-heuristic-based offloading task optimization in mobile edge computing. *Int. J. Distrib. Sens. Netw.* **2021**, *17*, 15501477211023021. [[CrossRef](#)]
23. Keshavarznejad, M.; Rezvani, M.H.; Adabi, S. Delay-aware optimization of energy consumption for task offloading in fog environments using metaheuristic algorithms. *Clust. Comput.* **2021**, *24*, 1825–1853. [[CrossRef](#)]

24. Xu, R.; Wang, Y.; Cheng, Y.; Zhu, Y.; Xie, Y.; Sani, A.S.; Yuan, D. Improved Particle Swarm Optimization Based Workflow Scheduling in Cloud-Fog Environment. In *Business Process Management*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 337–347.
25. Zhang, J.; Qi, L.; Yuan, Y.; Xu, X.; Dou, W. A Workflow Scheduling Method for Cloudlet Management in Mobile Cloud. In Proceedings of the 2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI), Guangzhou, China, 8–12 October 2018; pp. 932–937.
26. Lakhan, A.; Xiaoping, L. Energy Aware Dynamic Workflow Application Partitioning and Task Scheduling in Heterogeneous Mobile Cloud Network. In Proceedings of the 2018 International Conference on Cloud Computing, Big Data and Blockchain (ICCBB), Fuzhou, China, 15–17 November 2018; pp. 1–8.
27. Zhang, G.; Cheng, S.; Shu, J.; Hu, Q.; Zheng, W. Accelerating breadth-first graph search on a single server by dynamic edge trimming. *J. Parallel Distrib. Comput.* **2018**, *120*, 383–394. [[CrossRef](#)]
28. Tsirlis, K.; Lagani, V.; Triantafillou, S.; Tsamardinos, I. On scoring maximal ancestral graphs with the max–min hill climbing algorithm. *Int. J. Approx. Reason.* **2018**, *102*, 74–85. [[CrossRef](#)]
29. Tehreem, K.; Javaid, N.; Bano, H.; Ansar, K.; Waheed, M.; Butt, H. A cloud-fog based environment using beam search algorithm in smart grid. In *International Conference on Network-Based Information Systems*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 661–672.
30. Zhou, X.; Zhang, G.; Sun, J.; Zhou, J.; Wei, T.; Hu, S. Minimizing cost and makespan for workflow scheduling in cloud using fuzzy dominance sort based HEFT. *Future Gener. Comput. Syst.* **2019**, *93*, 278–289. [[CrossRef](#)]
31. Lee, J.; Lee, J. Juice Recipe Recommendation System Using Machine Learning in MEC Environment. *IEEE Consum. Electron. Mag.* **2020**, *9*, 79–84. [[CrossRef](#)]
32. Lee, J.; Lee, J. Hierarchical mobile edge computing architecture based on context awareness. *Appl. Sci.* **2018**, *8*, 1160. [[CrossRef](#)]
33. Lee, J.; Kim, J.W.; Lee, J. Mobile personal multi-access edge computing architecture composed of individual user devices. *Appl. Sci.* **2020**, *10*, 4643. [[CrossRef](#)]
34. Lee, J.; Kim, D.; Lee, J. Zone-based multi-access edge computing scheme for user device mobility management. *Appl. Sci.* **2019**, *9*, 2308. [[CrossRef](#)]
35. Kasi, S.K.; Kasi, M.K.; Ali, K.; Raza, M.; Afzal, H.; Lasebae, A.; Naeem, B.; Ul Islam, S.; Rodrigues, J.J. Heuristic edge server placement in industrial internet of things and cellular networks. *IEEE Internet Things J.* **2020**, *8*, 10308–10317. [[CrossRef](#)]