Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 5667012, 11 pages
https://doi.org/10.1155/2022/5667012

Research Article

SFDWA: Secure and Fault-Tolerant Aware Delay Optimal Workload Assignment Schemes in Edge Computing for Internet of Drone Things Applications

Abdullah Lakhan,[1] **Mohamed Elhoseny**,[2,3] **Mazin Abed Mohammed**,[4] **and Mustafa Musa Jaber**[5,6]

[1]*College of Computer Science and Artificial Intelligence, Wenzhou University, Wenzhou 325035, China*
[2]*College of Computing and Informatics, University of Sharjah, UAE*
[3]*Faculty of Computers and Information, Mansoura University, Egypt*
[4]*College of Computer Science and Information Technology, University of Anbar, Anbar 31001, Iraq*
[5]*Department of Computer Science, Dijlah University College, Baghdad, Iraq*
[6]*Department of Medical Instruments Engineering Techniques, Al-Farahidi University, Baghdad 10021, Iraq*

Correspondence should be addressed to Mazin Abed Mohammed; mazinalshujeary@uoanbar.edu.iq

Received 11 November 2021; Accepted 28 January 2022; Published 25 February 2022

Academic Editor: SK Hafizul Islam

The number of automobiles has rapidly increased in recent years. To broaden inhabitant's travel options, push transportation infrastructures to their limitations. With the rapid expansion of vehicles, traffic congestion and car accidents are all common occurrences in the city. The Internet of drone vehicle things (IoDV) has developed a new paradigm for improving traffic situations in urban areas. However, edge computing has the following issues such as fault-tolerant and security-enabled delay optimal workload assignment. The study formulates the workload assignment problem for IoV applications based on linear integer programming. The study devises the fault-tolerant and security delay optimal workload assignment (SFDWA) schemes that determine optimal workload assignment in edge computing. The goal is to minimize average response time, which combines network, computation, security, and fault-tolerant delay. Simulation results show that the proposed schemes gain 15% optimal workload assignment for IoV application compared to existing studies.

1. Introduction

The usage of vehicle transport in the workload has been growing for different purposes [1]. There are different types of carriers used to achieve other goals, such as cars, trucks, buses, railways, and drones [2]. But for the last decade, the usage of drone technologies has been increasing for different tasks day by day [3]. Drones are aircraft machines that fly in the sky to achieve various tasks. There are two significant types of drone aircraft, semiautonomous and full autonomous [4]. In the semiautonomous, the drone is handled by the remote control, which is operated by the ground-level human [5]. At the same time, self-autonomous is a fully automated machine that can handle the device itself without the interaction of any human being or remote from the ground level [6].

Recently, with the emerging development in cloud computing, different cloud models are introduced to handle the Internet of drone vehicle things by the servers [7]. For instance, ground-level services are integrated at other base stations along with fog nodes and edge nodes. The fog and edge nodes are cloud models which brought remote cloud services at the edge of base stations. The goal is to handle the Internet of drone vehicles (IoDV) from servers with their mechanism instead of the human being from ground level [8]. The edge and fog clouds have different categories for IoDV vehicles; for instance, IoDV can use many cloud applications to achieve other practice objectives and daily life

objectives. The main advantage of edge computing and fog computing is to provide a scalable environment without caring about the resources for the data process and store during the execution of IoDV applications in the network [9].

The fundamental challenge of IoDV in edge computing is to schedule workloads with the minimum delay requirements. Whereas each workload has a specific deadline and delay requirements, their executions at edge computing are critical tasks for IoDV applications. Many efforts were done in different research approaches to minimize the delay of drone applications in edge computing. These studies [2, 5, 6, 8] presented the network delay optimal solutions for drone applications in edge computing and proposed their schemes in work. The mobility and migration delay is considered in the network delay during offloading and processing of workload in the edge-enabled network, which is placed at the different locations. The computational delay for drone applications was also investigated in [1–3, 7] to achieve the workload execution in edge computing. The offloading of data from resource-constraint drone devices allows running applications to the available edge computing for the processing in the network. However, many other research challenges exist in the literature work due to offloading [9–11].

However, many research challenges exist in the existing delay optimal workload assignment strategies for drone applications. The existing studies only focused on drone applications' communication delay and computations delay. These delays did not handle any security and failure issues of applications in the edge computing network. Due to heavy workloads, the untrusted network has big security and failure due to malware attacks and scarcity of the resources at edge computing. Each workload has the particular deadline; the computational delay and network delay are sufficient to optimize but need to consider more delay types of drone applications in the edge computing.

In the paper, the study has the motivation to consider more delays, ensuring the security and failure of drone applications at heterogeneous edge computing. The study solves the following research questions. (i) The study optimizes the four types of delays in the proposed work: network delay, computational delay, security delay, and failure delay in the problem. (ii) The study optimizes the resource efficiency and meets the deadline of all workloads in edge computing. (iii) The study will design the simulation in which all proposed solutions can easily cooperate with edge computing.

The paper makes the following contribution to the work.

(i) The study drives the security and failure delay aware workload assignment (SFDWA) greedy metaheuristic in which the problem is to be solved with the different heuristics. That is called the dynamic divided and conquer programming model. The goal is to minimize all delay types of drone applications in edge computing and execute them under their deadlines

(ii) The work designs the architecture in which the process of the applications can show in terms of beginning and execution under different components as shown in Figure 1

(iii) In this work, there is design of the delay optimal mathematical model for the drone application in edge computing with the applications and node constraints

The manuscript consists of different sections from beginning to end. Section 2 shows all existing methods' strengths and research efforts for the delay-enabled applications. Section 3 defines the proposed architecture and mathematical models of drone applications in edge computing. Section 4 shows the processing model of the SFDWA metaheuristic for the considered problem. Sections 5 and 6 show the simulation process and conclusion part of the study.

2. Related Delay Optimal IoDV Schemes

Many promising solutions suggested metaheuristic-enabled methods to solve the workload assignment problem of drone applications in the homogeneous and heterogeneous edge cloud network. However, the Internet of drone vehicle things (IoDV) requires a challenging environment that consists of different drone sensors, wireless technologies, and edge cloud computing. For the simple transport applications, VANET and MANET are the famous platform to achieve the optimal workload assignment and have different dynamic heuristics that meet the deadline of applications. However, MANET and VANET cannot meet the requirements of IoDV applications in edge computing. Therefore, this session only discusses the closely related studies which solve the workload assignment problem of IoDV applications in edge computing as shown in Table 1.

In these studies [1–4], the authors suggested the genetic algorithm-based solutions for the drone applications considering both semiautonomous and fully autonomous case studies in the adaptive environment. These studies integrated the machine learning-enabled local search inside the genetic algorithm and minimized the studies' communication delay and computation delay. The study [5] suggested the IoDV application enabled a secure scheme where tasks are encrypted and decrypted at the edge cloud. The RSA-based encryption and decryption enabled IoDV-enabled method in mobility edge computing suggested in the network [2]. The computational delay optimal [6] schemes are suggested for IoV application in the edge cloud network. The goal is to reduce the processing delay of applications in edge computing. The network delay enabled IoV applications aware schemes suggested in [7, 8]. The goal is to minimize end to end network delay of applications. The RSA-based authentication and authorization for IoV application in distributed edge computing are devised in [9, 11–16].

As shown in Table 1, the existing studies focused on delay during workload assignment in edge computing. Table 1 demonstrates the difference between existing and the proposed work in edge computing during workload assignment problems. However, delay due to security and fault tolerant with network and computational delays is widely ignored in the literature's state as mentioned earlier of art. The work-study solves the workload assignment problem of IoV applications by considering the security and

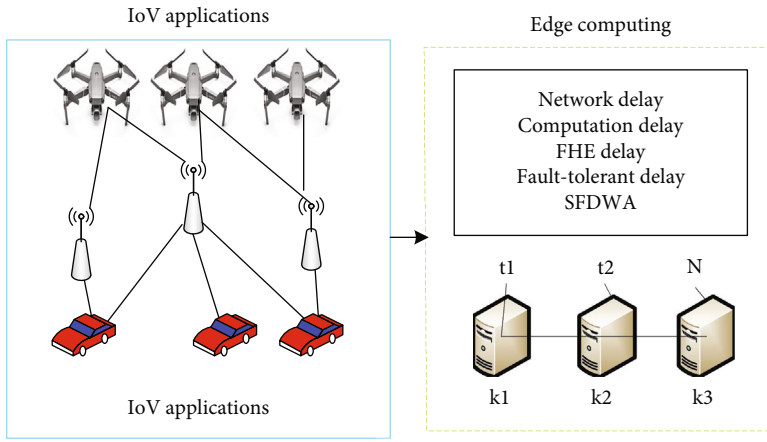

Figure 1: IoV architecture based on SFDWA schemes.

Table 1: Existing delay optimal workload assignment schemes for IoDV.

Research	IoV application	Security delay	Fault-tolerant delay	Network delay	Computation delay	Method
[1]	Traffic guidance	No	No	Yes	Yes	ILP greedy
[2]	Safe navigation	No	No	Yes	Yes	MILP greedy
[3]	Signal control	No	No	Yes	Yes	Knapsack greedy
[4]	Crash prevention	No	No	Yes	Yes	Iterative greedy
[5]	Toll collection	No	No	Yes	Yes	GA
[6, 7]	Traffic monitoring	No	No	Yes	Yes	PSO
[8, 9, 11]	Map route	No	No	Yes	Yes	Ant colony
[12, 13]	Location finding	No	No	Yes	Yes	MILP greedy
[14–16]	Fire ticketing	No	No	Yes	Yes	Partitioning greedy
[10, 17, 18]	Vehicle searching	No	No	Yes	Yes	Dynamic greedy
[19, 20]	Radar finding	No	No	Yes	Yes	Knapsack
[12, 21–25]	Radar safety	No	No	Yes	Yes	PSO greedy
Proposed	Package delivery	No	No	Yes	Yes	Iterative greedy

fault-tolerant delays, including computation delay and network delay for IoDV applications in edge computing. This work is totally different from closely related existing studies [17–25].

With the best of the authors' knowledge, the existing only considered the communication delay and computational delay of IoDV applications in the homogeneous edge networks. In the paper, the study has the motivation to consider more delays, ensuring the security and failure of drone applications at heterogeneous edge computing. The study solves the following research questions. (i) The study optimizes the four types of delays in the proposed work: network delay, computational delay, security delay, and failure delay in the problem. (ii) The study optimizes the resource efficiency and meets the deadline of all workloads in edge computing. (iii) The study will design the simulation in which all proposed solutions can easily cooperate with edge computing.

3. Problem Statement and Proposed System

The study formulates the delay optimal workload assignment at edge computing. There are different types of workload delays considered in the study. For instance, network delay, computation delay, security delay, and fault-tolerant delay are considered in this work in the edge computing for the workload. However, existing studies [1, 5, 8, 12] only felt the network delay and computational delay, which cannot be obtained from the security and failure of workload in their models. The study devises the SFDWA scheme-enabled system, which consists of different components as shown in Figure 1. The Internet of vehicle applications consists of fine-grained tasks with additional delay requirements such as network delay, security, computational delay, and fault-tolerant delay. All the jobs (e.g., tasks), e.g., $t = 1 \in N$, are independent, and each task has a deadline. On the other hand, edge computing consists of heterogeneous edge servers such as $k = 1, \in K$. The study devises the secure failure delay optimal workload assignment (SFDWA), consisting of different of network, computational, FHE (fully homomorphic encryption) delay schemes to execute workload in the heterogeneous edge nodes.

3.1. Problem Formulation. The study considered the T different types of IoDV applications in the considered problem. From T, each workload t has particular workload w_t and

execution deadline deadline_t. The study assumes the K heterogeneous edge nodes geographically distributed in 10 kilometers with the fixed environment. From K heterogeneous nodes, the edge node k has CPU speed ζ_k and resource limitation ε_k. All the applications connect to the edge nodes via B number of fixed base stations, where b from B only supports a fixed number of t workload requests from IoDV in the heterogeneous edge networks. The study describes the delay optimal as the partitioned combinatorial convex optimization problem as the PT. At the same time, the PT divides into four subproblems: security delay, network delay, computational delay, and failure delay.

3.1.1. Security Fully Homomorphic Delay. The study suggests a lightweight security scheme based on fully homomorphic additive encryption to minimize the security delay. It is an application requirement that all workloads encrypt and decrypt at the local devices and share cipher data to the edge nodes for computation and storage. Therefore, the study assumes that a D number of drones can encrypt and decrypt data locally inside applications T. The drone d has CPU processing capability p_d and determines the encryption and decryption in the following way, whereas $x_{k,t}$ and $y_{b,t}$ are the binary assignment to the base stations and edge nodes

$$\text{Encryption}_{tw} = \frac{w_t}{p_d} \times \text{Public} - \text{Key} \times \text{mod} \times x_{t,k}, y_{b,t}. \quad (1)$$

Equation (1) determines the encryption of workload w_t of application t with drone d based on 256-bit primary key with the additive module. The workload can decrypt on the same drone after execution in the following way

$$\text{Decryption}_{tw} = \text{Encryption}_{tw} \times \text{Private} - \text{Key} \times \text{mod}. \quad (2)$$

Equation (2) determines the decryption of workload w_t based on private key at the drone d. Therefore, the total security delay of all applications T on all drones D is determined in the following way:

$$\text{Security} - \text{Delay} = \sum_{t=1}^{T} \sum_{d=1}^{D} \text{Encryption}_{tw} + \text{Decryption}_{tw}. \quad (3)$$

Equation (3) determines the total security delay in the problem.

3.1.2. Drone Offloading Network Delay. All workloads are initially locally encrypted, and then, an offloader inside the drone offloads applications to the available base stations. The drone cannot directly offload all workloads to the edge nodes because all edge nodes are implemented at the base stations. Therefore, drone offload workloads to the base station and network delay are determined in the following way:

$$N\text{Delay} = \sum_{b=1}^{B} \sum_{t=1}^{T} \frac{(S/N) + \left(\text{Encryption}_{tw}/B_{\text{up}}\right)}{R} \times \text{status} = \text{ON} \times y_{b,t}. \quad (4)$$

In equation (4), $y_{b,t}$ is the binary assignment where $y_{b,t} = 1$ means the status of network is good; otherwise, it becomes 0, and S/N is the inference S and noise N with the error R and upload bandwidth up with the encrypted workloads from T to t_w. Equation (4) determines the network delay, if the network status becomes on and availability of base stations is optimal inside networks.

3.1.3. Heterogeneous Computational Delay. The workload assignment is to be done on the heterogeneous edge nodes in the following way.

$$C\text{Delay} = \sum_{k=1}^{K} \sum_{t=1}^{T} \frac{\text{Encryption}_{tw}}{\zeta_k} \times x_{k,t} \times \text{status} = \text{ON}. \quad (5)$$

Equation (5) determines allocation of encrypted workload to the optimal edge node k when it has binary assignment $x_{k,t} = 1$; otherwise, it becomes 0.

3.1.4. Failure of IoDV and Edge Delay. The failure can be identified by the status of work when process status becomes off, and then, scheduler will search another node within the deadline of workload in the edge node network.

$$\text{Failure} = \sum_{k=1}^{K} \sum_{t=1}^{T} \sum_{b=1}^{B} RT \times y_{b,t} \longleftarrow \text{status} = \text{OFF} \& x_{k,t} \longleftarrow \text{status} = \text{OFF}. \quad (6)$$

Equation (6) determines the failure status when it becomes OFF and recover time RT determined in the following way.

$$\text{RT} = \sum_{k=1}^{K} \sum_{t=1}^{T} \sum_{b=1}^{B} \text{RT} \longleftarrow \text{encryption}_{tw} \longleftarrow \text{status} = \text{ON} \leq \text{deadline}_t. \quad (7)$$

Equation (7) determines recovery time of the workload when it becomes ON in the network.

The problem PT is mathematically optimized in the following way.

$$\begin{aligned} \min\text{PT} = {} & \text{Security} - \text{Delay} + N\text{Delay} + C\text{Delay} \\ & + \text{Failure}, \quad \forall t = 1, \cdots, T. \end{aligned} \quad (8)$$

Equation (8) shows the objective function of the all workloads in the edge nodes network, subject to

$$\sum_{k=1}^{K} \sum_{t=1}^{T} \sum_{b=1}^{B} w_t \leq \varepsilon_k. \quad (9)$$

Equation (9) determines that all the workloads can execute under the available resources in the edge networks.

$$\sum_{k=1}^{K}\sum_{t=1}^{T}\sum_{b=1}^{B}\frac{w_t}{\zeta_k} \leq \text{deadline}_t. \tag{10}$$

Equation (10) determines that all the workloads execute under their deadlines in the edge networks.

$$\sum_{t=1}^{T} y_{b,t} = 1,\forall,b = 1, \cdots, B. \tag{11}$$

Equation (11) determines that each base station can offload one workload at a time.

$$\sum_{t=1}^{T} y_{k,t} = 1,\forall,k = 1, \cdots, K. \tag{12}$$

Equation (12) determines that each edge node can execute one workload at a time.

4. Proposed Algorithm SFDWA

The study solves the combinatorial convex optimization problem PT into partitioned subproblems called dynamic programming problems for IoDV applications in heterogeneous edge nodes. The study devises the secure failure delay workload assignment (SFDWA) metaheuristic; it has the following subheuristics: solving the subproblems into optimal solutions. The SFDWA is a complete metaheuristic consisting of different components for each IoDV application in the edge node networks. Each part is a heuristic independent of another element, which means the optimization of each element is calculated individually in the metaheuristic. In this way, we can control the application deadline performances with each component in the problem PT. Algorithm 1 is a metaheuristic which consists of different processes in the algorithm.

4.1. Security Delay-Enabled Component. Algorithm 2 takes the input of all applications in terms of their workloads and encrypts them on the local drone machine before offloading to the base station. The study allows drones to encrypt and decrypt the data, and base station and edge nodes can process data computation only on the cipher data. This work is totally different from existing studies; in this work, the cipher data process on the base station and edge node without decrypting them in the network. The data encrypted based on 256-bit primary key generated based on advanced standard encryption scheme in work. The two random large integers are exploited in public and private keys at the drone device, where data is encrypted and decrypted with both public and private keys. All the encrypted are offloaded to the base station for further processing.

4.2. Network Delay-Enabled Component. Algorithm 3 takes the encrypted data of all applications in terms of their workloads if the status of all base stations is equal to 1 not zero. The status ON and OFF shows that the availability of base stations is sufficient and bandwidth can offload workload to the edge nodes for the processing.

4.3. Computational Processing Delay. The study devises the combinatorial convex-enabled dynamic scheduling scheme where objective function PT optimizes the heterogeneous edge nodes. At the same time, the deadline and resource limitations are the convex set in the problem. The scheduler sorts all encrypted workloads into their deadline order to minimize the computational delay. The minor deadline enabled workloads to get high priority in the scheduling queue. The scheduler has two phases: topological

Input: $b = 1, \cdots, B, k = 1, \cdots, K,$
$t = 1, \cdots, T$, minPT.
1 B**egin**
2 Status = 0, 1 = ON or OFF;
3. Component 1 secure mechanism;
4. Component 2 offloading networking delay scheme;
5. Component 3 computing delay scheme;
6. Component 4 failure recovery mechanism;
7. Optimize status PT* ⟵ = 1, b, k, t;
8. End main;

Algorithm 1: SFDWA metaheuristic algorithm.

Input: $t = 1, \cdots, T, d = 1, \cdots, D, k = 1, \cdots, K$
1 B**egin**
2 PK ⟵ PrimaryKey;
3 PV ⟵ PrivateKey;
4 Encryption process;
5 **for each** (tw ⟵ $t1$)
6 Determined encryption time based on equation (1);
7 Generate large random number p, q;
8 mod ⟵ PK ⟵ 256 *bits*;
9 Encryption_{tw} ⟵ t_w ⟵ $t1 \times d1 + p \times q \times \text{mod}$;
10 Decryption process;
11 **for each** (Encryption_{tw} ⟵ t_w ⟵ $t1 \times k1$) **do**
12 $\text{Decryption}_{tw}[t1, T]$ ⟵
Encryption_{tw} ⟵ PV;
13 End main;

Algorithm 2: Fully homomorphic encryption scheme.

Input: $\text{Encryption}_{tw}[t, T], b \in B$
1 B**egin**
2 (b ⟵ Status = 1) **then**
3 b ⟵ status = ON:
4 Determined the available B_{UP}:
5 Determined the network delay based on equation (4);
6 Make the initial assignment based on $y_{b,\text{encryption}_{tw},t} = 1$;
7 Offload $\text{Encryption}_{tw}[t1, T]$;
8 End main;

Algorithm 3: Network delay component.

Input: MQW$[t, T]$, RQ$[k, K]$
1 **Begin**
2 **for each** (MQW$[t, T]$ & RQ$[k, K]$) **do**
3 Determined the computational delay based on equation (5);
4 Make the initial assignment based on equations (11) and (12);
5 Determined the deadline and resources based on (10) and (9);
6 **if** (Encryption$_{tw} \longleftarrow t \longleftarrow k \leq$ deadline$_t$) **then**
7 PT $\longleftarrow x_{k,t}$;
8 PT = Security − Delay + NDelay + CDelay + Failure;
9 Schedule $x_{k1,t1} \in T \longleftarrow$ Status == ON based on equation (8)
10 Optimize PT;
11 **if** ($x_{k,t} \longleftarrow$ Status == 0) **then**
12 Schedule $x_{k,t} \longleftarrow$ Status == OFF based on equation (8);
13 Apply local search;
14 **if** (PT $\longleftarrow k1 \leq$ PT $\longleftarrow k2$) **then**
15 replace $f(\text{PT}^*) \longleftarrow f(\text{PT})$;
16 Call checkpointing;
17 Reschedule schedule $x_{k2,t1} \in T \longleftarrow$ Status == ON based on equation (8);
18 End main;

Algorithm 4: Computational processing delay component.

prioritizing and sharing parallel execution of workloads in edge nodes. Initially, all the workloads arrived randomly in the M/M/1-PS sharing edge nodes queue where there is no waiting delay. The M/M/1-PS Queue-Workload (MQW) [14] sorts all workloads based on their deadlines. For instance,

$$\text{MQW}[t, T] = \text{Sorting}\left(\sum_{t=1}^{T} \text{Encryption}_{tw,t,T} \longleftarrow \text{deadline}_t\right). \tag{13}$$

Equation (13) sorts all the workloads into their deadline before being executed to the edge nodes. All the edge nodes sort according to their availability of the resources and high speed in the resource queue (RQ) as determined in

$$\text{RQ}[k, K] = \text{Sorting}\left(\sum_{k=1}^{K} \varepsilon_k \& \zeta_k\right). \tag{14}$$

High-speed nodes have less delay and should have sufficient resources to run the scheduled workload in the edge networks.

Algorithm 4 is a greedy algorithm that schedules all encrypted workloads based on their deadlines and available resources of edge nodes. Each encrypted workload must be executed its deadline and within available resources with status == 1 and status == ON. If the failure occurs in a particular edge node $k1$, then the status becomes status == 0 and status == OFF. The checkpointing enabled delay optimal policy call and search another edge node with the optimal objective function $f(\text{PT}^*) \longleftarrow k2 \longleftarrow f(\text{PT}) \longleftarrow k1$ which replaces the failure objective function to another node objective function in the network. This way, we can optimize all workload execution in terms of the objective function in the network.

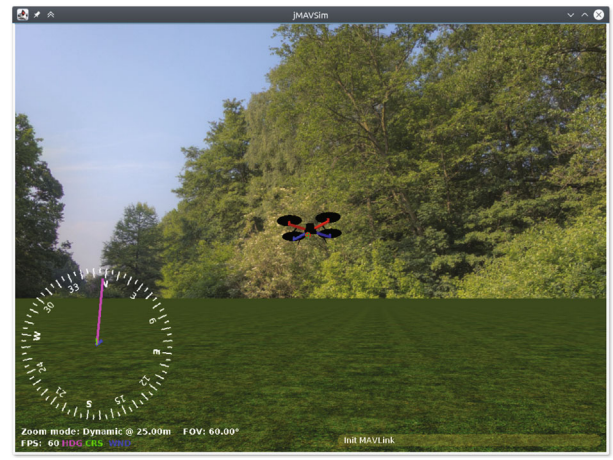

Figure 2: jMAVSim simulation tool enabled IoDV experimental environment.

5. Performance Evaluation

In the evaluation part, the study implemented SFDWA algorithm framework, and baseline 1 and baseline 2 approaches in available jMAVSim simulation tool enabled IoDV experimental environment.

The study designed the simulation environment on the available tool jMAVSim on the link "https://github.com/PX4/jMAVSim." Figure 2 shows the jMAVSim simulation environment with the IoV task application in the distributed system. This simulation environment is already implemented in our previous manuscript [16] and is widely deployed in the simulation environment with the blockchain and scheduling techniques.

Table 2: IoDV package delivery application dataset and sensors.

N	w_t (MB)	deadline$_t$ (seconds)	B	K (core)	ε_k (GB)	b_{bw} (MBPS)	$D \longleftarrow$ Mavic IoDV drone sensors
$t1$	10	300	500	1	High core	10	Drone
$t2$	10	300	500	1	High core	10	Drone
$t30$	10	300	500	1	High core	10	Drone
$t100$	10	300	500	1	High core	10	Car
$t200$	10	300	500	1	High core	10	Car
$t250$	10	300	500	1	High core	10	Car
$t300$	10	300	500	1	High core	10	Drone

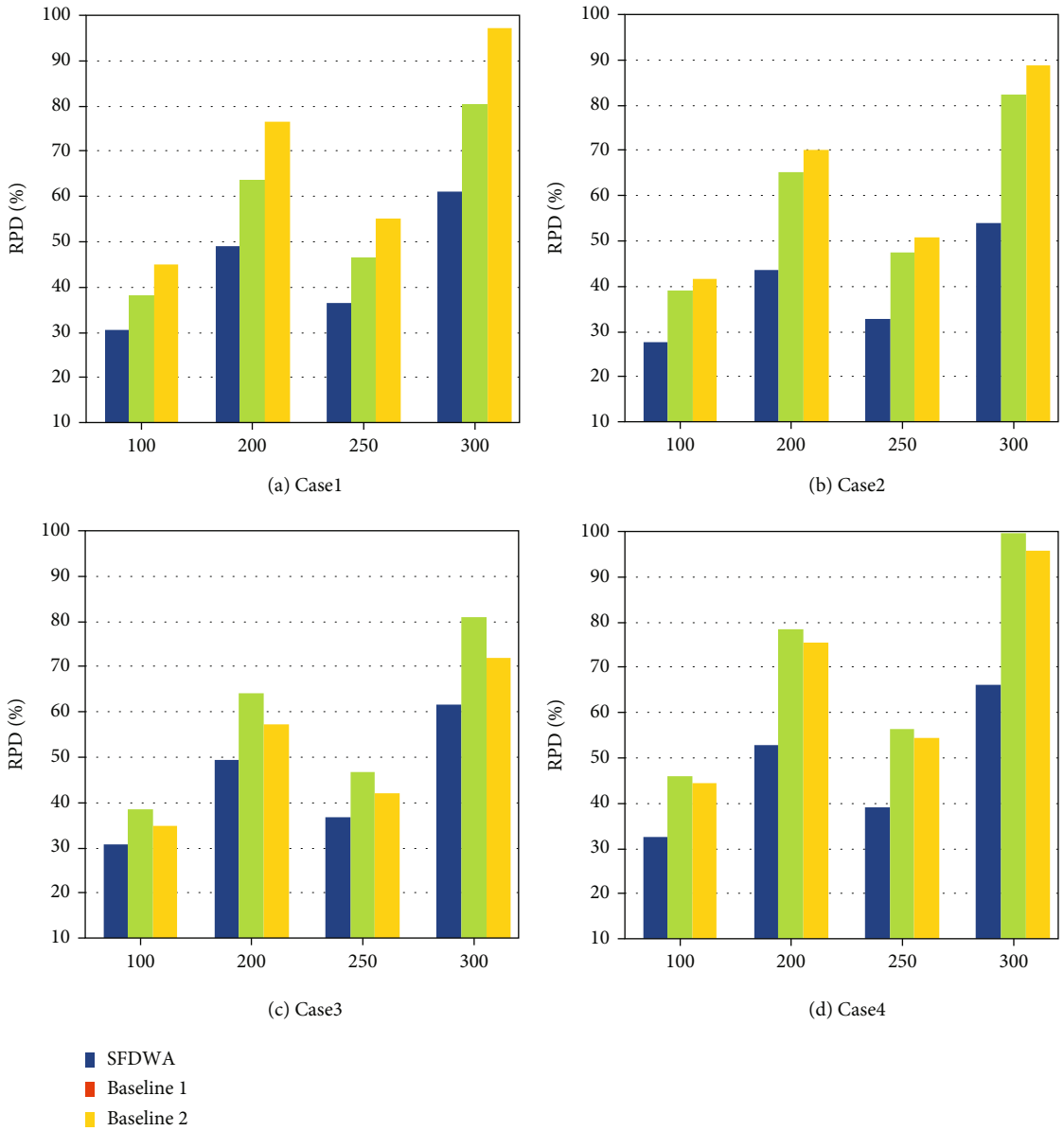

Figure 3: Network delay in different cases.

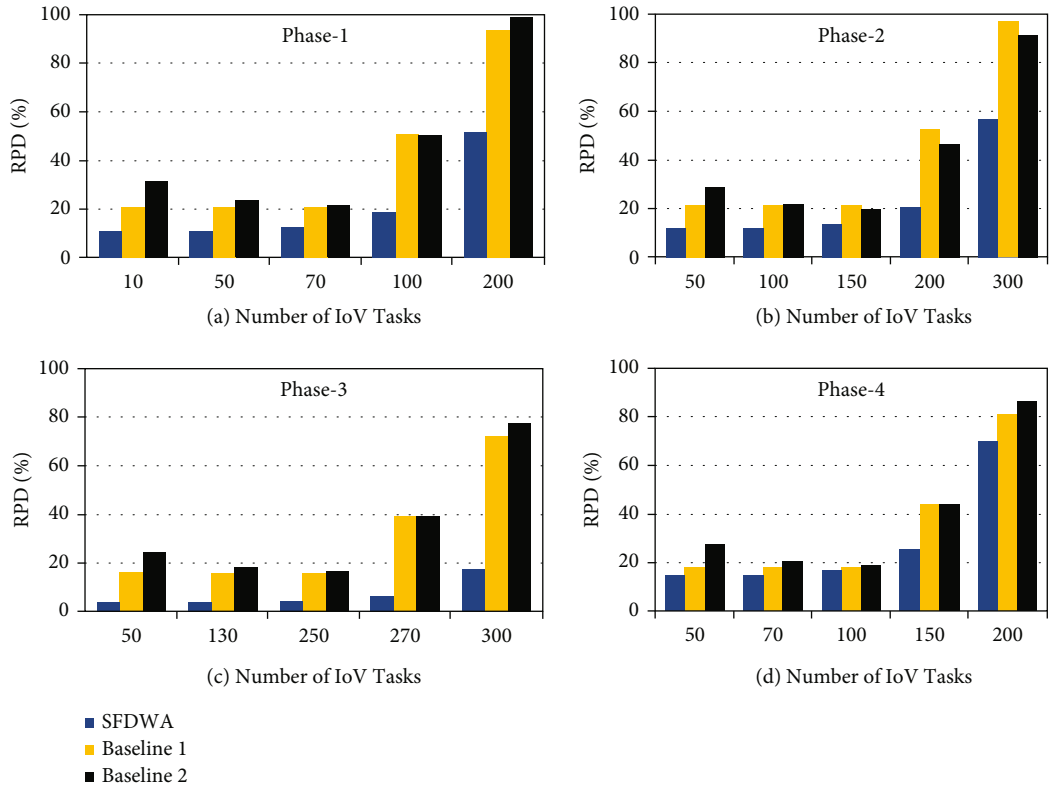

Figure 4: Workload assignment performances in all phases.

5.1. IoDV Application Dataset. The dataset parameters of the IoDV applications are shown in Table 2 implemented in the simulation. The dataset detail is available in [16] study. The study exploited the real dataset for the experiment, which is obtained and detailed in [16] site.

5.2. Closely Related Baseline Approaches. The study is going to make an experimental comparison of the proposed solution with other similar works, whereas baseline 1 [3, 5, 7, 9, 16] widely implemented delay optimal drone applications in the edge computing network. These studies considered network delay and computational delay for workload assigning edge computing, whereas baseline 2 is widely deployed in [1, 4, 6, 11, 13] studies. These studies tried to minimize end to end delay of applications with minimum consumption of latency for IoDV applications.

5.3. Performance Metrics. The study exploits statistics to evaluate the results of the proposed schemes with relative percentage deviation (RPD) as follows.

$$\mathrm{RPD}(\%) = \frac{\mathrm{PT}^* - \mathrm{PT}}{\mathrm{PT}^*} \times 100\%. \tag{15}$$

Z^* displays the optimal workload assignment of IoV applications.

5.4. Result Discussion. This section discusses the obtained results of the methods for the considered problem with different metrics as follows in different subsections.

5.5. Network Delay in Different Cases. The study considered the different cases for the network delay in the simulation part based on the objective function. If S is 20 dB, b_{bw} is 4 kHz for the network delay. Then, we discussed it with different cases and evaluated the performances in the simulation. Case 1: bandwidth = 15000 $N(1 + S = 200) = 4000$ $R(101) = 36.63$ kbit/s. Case 2: bandwidth = 7000 $S(1 + 100)$ $= N = 7000$ $R(101) = 46.63$ kbit/s. Case 3: bandwidth = 2000 $S(1 + 100) = N = 2000$ $R(101) = 16.63$ kbit/s. Case 4: bandwidth $S(1 + 100) = N = 4000$ $R(101) = 26.63$ kbit/s. These different values are implemented in the simulation config file for the simulation.

Figure 3 shows that, with different IoV tasks, the network delay with different cases could be changed in RPD%. Therefore, in Figures 3(a)–3(d), the proposed SFDWA outperformed all existing baseline approaches in terms of network delay in RPD%.

5.6. Comparison of Workload Assignment Algorithms. Figure 4 shows the performances of all phases with the proposed scheme and baseline approaches. The RPD% of all existing baseline approaches is shown in the evaluation result. Figure 4(a) shows the RPD% of the network delay for IoV applications with a random number of tasks in edge computing. Figure 4(a) shows the RPD% of the proposed SFDWA outperformed all existing schemes due to many reasons. The first reason is that all existing only focused on network delay without considering the availability of edge computing resources. These baseline 1 and baseline 2 only assumed the network delay resource and ignored the

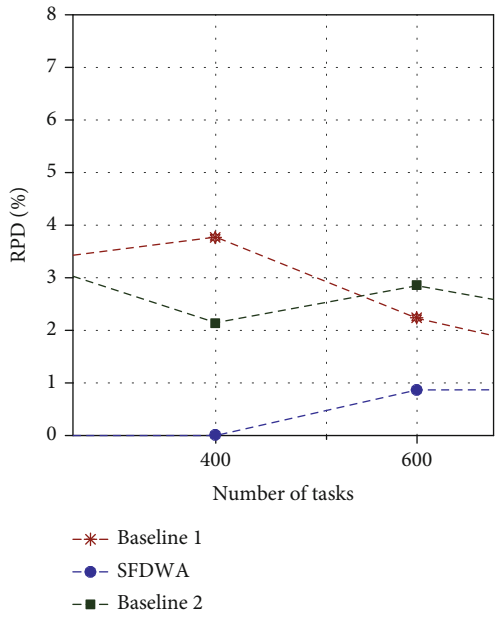

Figure 5: RPD% performances of IoDV tasks with network delay.

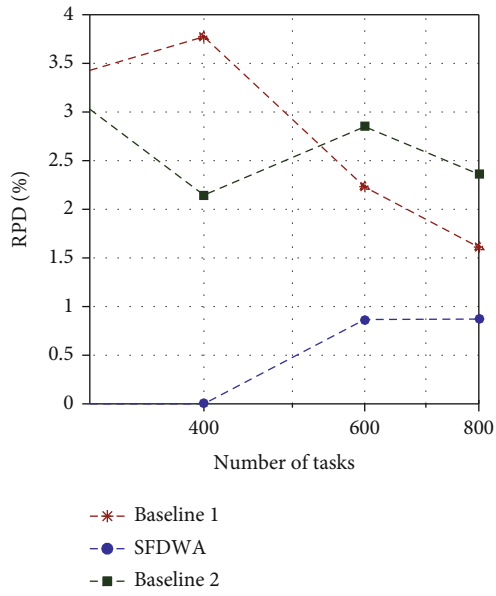

Figure 6: RPD% performances of network delay and computational delay for IoDV workloads.

schemes' security and fault-tolerant edge computing delay. Figure 4(b) shows the computational delay and RPD% could be improved whenever the network and computational delay are jointly optimized simultaneously in the edge computing node. Another reason is that we determined the resources of computing nodes in advance by implementing resource profiling technologies at the compile time. Figures 4(c) and 4(d) show the SFDWA outperformed all existing baseline approaches. The main reason is that all current studies in phase 1, phase 2, phase 3, and phase 4 only considered the network delay and computational delay without considering the security delay and fault tolerant in the edge computing network. Another reason is that all existing schemes did not consider the deadline of application tasks; therefore, minimizing the delay for IOV applications without deadline constraint is ineffective in edge computing. The resource leakage could be possible if the number of requests exceeds the resource limits and be incurred with longer end to end delay. Therefore, the proposed scheme SFDWA has a control from offloading to execution with both security and deadline constraints and improved the overall RPD% of different phases in edge computing.

Figure 5 shows the RPD% performances of the computational delay for IoV tasks by applying the proposed method and baseline approaches. The study only considers case 1 in this experimental part, where the computational delay was determined before offloading to edge computing with the related base stations. Still, SFDWA outperforms computational delay compared to existing baseline approaches because existing works only considered the case 1 situation where network availability is only considered and other parameters are widely ignored, as we discussed in different cases, the performance of network delay. Therefore, network delay is also a vital aspect of offloading workloads to edge computing.

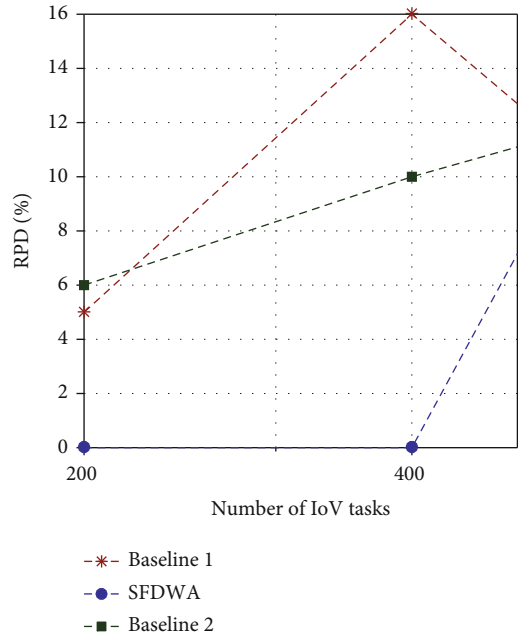

Figure 7: RPD% performances of IoDV workload with network delay.

Figure 6 shows the RPD% performances of computational delay and communication delay for IoV tasks in edge computing. The delay increases when both computational and communication delays are calculated in edge computing. However, the study exploited different cases and not allowed offload tasks to the weak signal networks and overloading edge nodes; in this way, low priority tasks can offload later and minimizes the computational delay and communication delay in the edge computing network. Therefore, SFDWA outperformed all existing baseline approaches in terms of computational delay and communication delay.

Figure 7 is the total delay including network delay, computational delay, security delay, and fault-tolerant delay and the impact of uncertainty delays can be analyzed in Figure 7. The ratio of delay with both baselines approaches growth compared to SFDWA. The main reason is that all existing baseline approaches only considered the computational delay and network delay in their schemes for IoV tasks in edge computing. Therefore, the proposed idea and SFDWA are optimal and effective in edge computing to control the different types of delay for IoDV workloads.

In Figure 7, there is a lot of impact of different delays according to requirements. For instance, security is the demand in the recent developed system, and fault tolerant is the backbone; therefore, these delays cannot be ignored in the edge computing for IoV applications. If we leave them, only considering the communication delay and computational delay, Figure 7 shows the impact of delays for IoDV workloads in the edge computing.

6. Conclusion

The study formulates the workload assignment problem for IoV applications based on linear integer programming. The study devised the fault-tolerant and security delay optimal workload assignment (SFDWA) schemes that determined optimal workload assignment in edge computing. The goal is to minimize average response time, which combines network, computation, security, and fault-tolerant delay. Simulation results show the proposed schemes gain 15% optimal workload assignment for IoV application in edge computing compared to existing studies.

In the future work, the study will discuss the mobility delay and migration delay in the current architecture to improve the mobility performance of the IoV applications.

Data Availability

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

Conflicts of Interest

The authors declare that there is no conflict of interest.

References

[1] A. Salama, A. M. Mostafa, S. Gunasekaran et al., "An agent architecture for autonomous uav flight control in object classification and recognition missions," *Soft Computing*, vol. 3, pp. 1–14, 2021.

[2] S. A. Mostafa, M. S. Ahmad, A. Mustapha, and M. A. Mohammed, "Formulating layered adjustable autonomy for unmanned aerial vehicles," *International Journal of Intelligent Computing and Cybernetics*, vol. 10, no. 4, pp. 430–450, 2017.

[3] Q. Fan and N. Ansari, "Workload allocation in hierarchical cloudlet networks," *IEEE Communications Letters*, vol. 22, no. 4, pp. 820–823, 2018.

[4] M. A. R. Dantas, P. E. Bogoni, and P. Jos'e De Freitas Filho, "An application study case tradeoff between throughput and latency on fog-cloud cooperation," *International Journal of Networking and Virtual Organisations*, vol. 23, no. 3, pp. 247–260, 2020.

[5] V. Chamola, C. K. Tham, S. Gurunarayanan, and N. Ansari, "An optimal delay aware task assignment scheme for wireless sdn networked edge cloudlets," *Future Generation Computer Systems*, vol. 102, pp. 862–875, 2020.

[6] A. Lakhan, M. A. Mohammed, A. N. Rashid et al., "Smart-contract aware ethereum and client-fog-cloud healthcare system," *Sensors*, vol. 21, no. 12, p. 4093, 2021.

[7] R. Deng, L. Rongxing, C. Lai, T. H. Luan, and H. Liang, "Optimal workload allocation in fog-cloud computing towards balanced delay and power consumption," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 1171–1181, 2016.

[8] S.-C. Wang, S.-C. Tseng, K.-Q. Yan, and Y.-T. Tsai, "Reaching agreement in an integrated fog cloud iot," *IEEE Access*, vol. 6, pp. 64515–64524, 2018.

[9] W. Zhang, Z. Zhang, S. Zeadally, H.-C. Chao, and V. C. M. Leung, "Masm: a multiple-algorithm service model for energy-delay optimization in edge artificial intelligence," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 7, pp. 4216–4224, 2019.

[10] A. Lakhan, M. A. Dootio, A. H. Sodhro et al., "Cost-efficient service selection and execution and blockchain-enabled serverless network for Internet of medical things," *Mathematical Biosciences and Engineering*, vol. 18, no. 6, pp. 7344–7362, 2021.

[11] C. Martín, D. Garrido, L. Llopis, B. Rubio, and M. Díaz, "Facilitating the monitoring and management of structural health in civil infrastructures with an edge/fog/cloud architecture," *Computer Standards & Interfaces*, vol. 81, article 103600, 2022.

[12] X. Guo, R. Singh, T. Zhao, and Z. Niu, "An index based task assignment policy for achieving optimal power-delay tradeoff in edge cloud systems," in *2016 IEEE International Conference on Communications (ICC)*, pp. 1–7, Kuala Lumpur, Malaysia, 2016.

[13] H. Chegini, R. K. Naha, A. Mahanti, and P. Thulasiraman, "Process automation in an iot–fog–cloud ecosystem: a survey and taxonomy," *IoT*, vol. 2, no. 1, pp. 92–118, 2021.

[14] H. Cao and M. Wachowicz, "An edge-fog-cloud architecture of streaming analytics for Internet of things applications," *Sensors*, vol. 19, no. 16, p. 3594, 2019.

[15] F. Al-Turjman, M. Z. Hasan, and H. Al-Rizzo, "Task scheduling in cloud-based survivability applications using swarm optimization in Iot," *Transactions on Emerging Telecommunications Technologies*, vol. 30, no. 8, article e3539, 2019.

[16] R. O. Aburukba, M. AliKarrar, T. Landolsi, and K. El-Fakih, "Scheduling Internet of Things requests to minimize latency in hybrid fog- cloud computing," *Future Generation Computer Systems*, vol. 111, pp. 539–551, 2020.

[17] A. Lakhan, M. A. Mohammed, O. I. Obaid, C. Chakraborty, K. H. Abdulkareem, and S. Kadry, "Efficient deepreinforcement learning aware resource allocation in SDN-enabled fog paradigm," *Automated Software Engineering*, vol. 29, no. 1, pp. 1–25, 2022.

[18] Y. Deng, Z. Chen, D. Zhang, and M. Zhao, "Workload scheduling toward worst-case delay and optimal utility for single-hop fog-iot architecture," *IET Communications*, vol. 12, no. 17, pp. 2164–2173, 2018.

[19] H. Mora, F. J. Mora Gimeno, M. T. Signes-Pont, and B. Volckaert, "Multilayer architecture model for mobile cloud computing paradigm," *Complexity*, vol. 2019, Article ID 3951495, 13 pages, 2019.

[20] X. Wang, W. Guo, W. Zhang, M. K. Khan, and K. Alghathbar, "Cryptanalysis and improvement on a parallel keyed hash function based on chaotic neural network," *Telecommunication Systems*, vol. 52, no. 2, pp. 515–524, 2013.

[21] A. Lakhan, M. A. Mohammed, S. Kadry, K. H. Abdulkareem, F. T. Al-Dhief, and C.-H. Hsu, "Federated learning enables intelligent reflecting surface in fog-cloud enabled cellular network," *Peer J Computer Science*, vol. 7, article e758, 2021.

[22] A. H. Sodhro, Z. Luo, A. K. Sangaiah, and S. W. Baik, "Mobile edge computing based qos optimization in medical healthcare applications," *International Journal of Information Management*, vol. 45, pp. 308–318, 2019.

[23] A. H. Sodhro, S. Pirbhulal, and V. H. C. De Albuquerque, "Artificial intelligence-driven mechanism for edge computing-based industrial applications," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 7, pp. 4235–4243, 2019.

[24] A. H. Sodhro, S. Pirbhulal, Z. Luo, and V. H. C. de Albuquerque, "Towards an optimal resource management for iot based green and sustainable smart cities," *Journal of Cleaner Production*, vol. 220, pp. 1167–1179, 2019.

[25] S. A. Hassan and Y. Li, "Medical quality-ofservice optimization in wireless telemedicine system using optimal smoothing algorithm," *E-Health Telecommunication Systems and Networks*, vol. 2, no. 1, pp. 1–8, 2013.