electronics

MDPI

Article

Multi-Layer Latency Aware Workload Assignment of E-Transport IoT Applications in Mobile Sensors Cloudlet Cloud Networks

Abdullah Lakhan [1,2], Mazhar Ali Dootio [1], Tor Morten Groenli [3,*], Ali Hassan Sodhro [4,5,6] and Muhammad Saddam Khokhar [1]

1 Research Lab of AI and Information Security, Department of Computer Science and Technology, Benazir Bhutto Shaheed University Lyari, Karachi 75660, Pakistan; abdullahrazalakhan@gmail.com (A.L.); mazharaliabro@bbsul.edu.pk (M.A.D.); saddam_khokhar@hotmail.com (M.S.K.)
2 College of Computer Science and Artificial Intelligence, Wenzhou University, Wenzhou 325035, China
3 Mobile Technology Lab., Department of Technology, Kristiania University College, 0107 Oslo, Norway
4 Department of Computer and System Science, Mid Sweden University, 831 25 Ostersund, Sweden; alihassan.sodhro@miun.se
5 Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518000, China
6 Electrical Engineering Department, Sukkur IBA University, Sukkur 65200, Pakistan
* Correspondence: tor-morten.gronli@kristiania.no

Citation: Lakhan, A.; Dootio, M.A.; Groenli, T.M.; Sodhro, A.H.; Khokhar, M.S. Multi-Layer Latency Aware Workload Assignment of E-Transport IoT Applications in Mobile Sensors Cloudlet Cloud Networks. *Electronics* **2021**, *10*, 1719. https://doi.org/10.3390/electronics10141719

Academic Editors: Dongkyun Kim, Qinghe Du, Mehdi Sookhak, Lei Shu, Nurul I. Sarkar, Jemal H. Abawajy and Francisco Falcone

Received: 18 June 2021
Accepted: 9 July 2021
Published: 17 July 2021

Abstract: These days, with the emerging developments in wireless communication technologies, such as 6G and 5G and the Internet of Things (IoT) sensors, the usage of E-Transport applications has been increasing progressively. These applications are E-Bus, E-Taxi, self-autonomous car, E-Train and E-Ambulance, and latency-sensitive workloads executed in the distributed cloud network. Nonetheless, many delays present in cloudlet-based cloud networks, such as communication delay, round-trip delay and migration during the workload in the cloudlet-based cloud network. However, the distributed execution of workloads at different computing nodes during the assignment is a challenging task. This paper proposes a novel Multi-layer Latency (e.g., communication delay, round-trip delay and migration delay) Aware Workload Assignment Strategy (MLAWAS) to allocate the workload of E-Transport applications into optimal computing nodes. MLAWAS consists of different components, such as the Q-Learning aware assignment and the Iterative method, which distribute workload in a dynamic environment where runtime changes of overloading and overheating remain controlled. The migration of workload and VM migration are also part of MLAWAS. The goal is to minimize the average response time of applications. Simulation results demonstrate that MLAWAS earns the minimum average response time as compared with the two other existing strategies.

Keywords: MLAWAS; Q-Learning; response-time; simulation; assignment

1. Introduction

These days, cloud computing-based smart applications are growing progressively to deal with different life activities. The applications are E-Healthcare, E-Banking, Augmented Reality and E-Commerce [1]. Cloud computing offers the abandonment of being ubiquitous and omnipresent in order to entertain the requests of applications. However, conventional cloud computing, located multiple hops away from E-Transport, and the offloading of the workload of applications, will suffer from long end-to-end latency. Cloudlet is an extension of cloud computing that brings computing resources to the edge of the communication network. Cloudlet is a latency-optimal paradigm and is easily connected with the network's base stations, controlled by the software-defined network. Offloading and workload assignment are two fundamental techniques in distributed cloudlet-based cloud computing networks where E-Transports send their workload for execution. Generally, it is called full offloading from E-Transport devices to the cloudlet-based cloud network [2].

There are many types of latency that exist in the cloudlet based cloud network; for instance, the communication latency, round-trip latency, process latency, and migration latency during offloading and workload assignment. These latency types greatly impact applications' performance during offloading and processes in the network. Therefore, multi-layer latency workload assignment has become a difficult problem in cloudlet-based cloud networks. Many efforts have been made to solve the latency optimal workload assignment in the cloudlet-based cloud network. For example, the authors of [1–3] suggested greedy-based workload assignment in the cloudlet-based cloud network, where process latency is taken into consideration. The authors of [4–7] suggested a dynamic programming-based workload assignment solution in a distributed cloudlet network where the dynamic changes of network contents are taken into consideration. The process latency and communication are considered for each workload before offloading and assignment to different computing nodes. Refs. [8–11] suggested a heuristics-based solution to optimize workload assignment and obtained a near-optimal solution for applications. The process delay and the communication were considered the constraints, and the average response time of each application was the objective function of the studies. The mobility aware offloading and the workload assignment, including process delay and communication, were formulated in the studies of [12–15]. These studies assumed that all communication networks and processing nodes always remain stable and optimal. However, this is unrealistic because communication networks and computing nodes within a dynamic environment, where E-Transports change their location, cannot remain optimal. At the same time, round-trip latency and migration latency during the migration of workload between nodes has been widely ignored in the literature. The suggested methods—based on a greedy approach to obtain optimal local search, and heuristics to obtain the global search for an optimal solution—cannot work in a dynamic environment in which many parameters change due to the mobility of E-Transports [16–19].

This paper formulates a multi-layer latency-aware workload assignment in the cloudlet-based cloud network. The objective is to minimize the average response time of all E-Transport applications. The average response time is determined by processing delay and communication delay (e.g., round-trip delay and migration delay) during offloading and workload assignment. The E-Transport mobility and the migration of workload are also considered to avoid load balancing and the overhead situation in the network. This study addresses the following research questions: (i) How do we design a resource-optimal, cloudlet-based cloud network that will not incur overloading and resource-constraint issues? (ii) How do we adopt dynamic arrival workloads based on the Poisson process in the network without waiting for the delay? (iii) How do we search for a local cloudlet that is optimal compared to other cloudlets for workload assignments? (iv) How do we adopt dynamic changes and obtain global search-based optimal solutions during mobility and the migration of workload in the network?

This study makes the following contributions to answer the questions stated above:

- Proposed System: Initially, this study proposes a mobile cloudlet-based cloud system where E-Transports can easily offload their coarse-grained applications. The mobile devices work as a thin client, where E-Transport IoT applications only offload their workload via a client cloudlet module. The cloudlet host is a distributed server-based network in which all servers are heterogeneous, and each workload is assigned a single virtual machine as a guest to run. The cloud remote is the conventional cloud that is the part of the system which supports the migrated workloads for further execution. The mobile cloudlet based cloud (MCBC) is a distributed system in which the workload can execute at different computing nodes during offloading and execution.
- To solve the workload assignment in a distributed MCBC network, this study proposes the MLAWAS framework, which consists of different schemes which solve the problem into different steps. The schemes are defined in their next specific points in detail.
- Initial Optimal Heuristic: This study proposes the improved Genetic Algorithm (IGA) based on a natural selection mechanism that mimics biological evolution to solve

constrained and unconstrained optimization problems. A workload population of individual solutions is updated repeatedly by the algorithm. An optimal solution reaches the best point in the workload population.

- Global Searching: For optimization, improved Genetic Algorithms are a global search technique. They are bad at hill-climbing. However, the potential for probabilistic hill-climbing is improved Simulated Annealing. Therefore, by adding a mutation operator such as Simulated Annealing and an adaptive cooling schedule, the two techniques are combined here to create an adaptive algorithm that has the merits of both Genetic Algorithms and Simulated Annealing.
- Local Greedy Searching: The greedy-based search is proposed to deal with local search and to find the optimal solution of any requested workload within available cloudlets. However, it is working in the same environment. The main goal of this method is to verify that all workloads assigned to computing nodes have optimal average response time solutions compared to others. This searching repeats iteratively until all workloads are assigned to all available component nodes.
- Q-Learning Aware Migration Technique: This study suggests a Q-Learning-based VM migration method where the current optimal solution being treated as a state is an improvement on the objective solution, and the workload to be migrated to benefit E-Transport benefits mobility. The Q-Learning method always adopts any dynamic changes of the environment and always chooses an optimal solution during the migration of the workload to maintain the workload and the overhead situation in the MCBC network.

The rest of the paper is planned as follows: Section 2 discusses the literature on the latency optimal workload assignment distributed network. Section 3 contains a mathematical model of how to define all kinds of delays with their constraints. Section 4 clarifies how we utilize the techniques chosen to lower the delays in conjunction with the mathematical model. Section 5 presents the simulation system and the results of the discussion of different schemes. Section 6 presents the summary of the study and the future direction of the research.

2. Related Work

The latency optimal workload assignment problem in the mobile cloudlet based cloud network (MCBC) has been widely studied in the literature. This problem is formulated as a linear integer problem, a greedy problem and a combinatorial problem by different researchers in the literature. Table 1 describes the existing problem formulation and the proposed schemes for the workload assignment problem in the MCBC network.

Table 1. Comparison among existing works and the proposed work.

Existing Works	[1–5]	[6–12]	[13–28]	**Proposed**
Objective	Task offloading	Task offloading	Task offloading	Task offloading
Constraints	Cost	none	none	Hybrid latency
Network Delay	✓	×	×	✓
Process Delay	×	✓	✓	✓
Migration Delay	×	×	×	✓
Fine grained Offloading	Thread level	Code level	Application partitioning	VM overlay
Application scenario	Social media	3D Gaming	Social media	Workloads
Algorithm Complexity	Exponential	Exponential	No guarantee	Assignment guarantee
Performance	Near-optimal	No guarantee	Local optimum	Global optimal
Method	ILP Greedy	NP-Hard Heuristics	HMM	Q-Learning

Nouha et al. [1] formulated latency and flexible workload assignment distributed cloudlet networks. The goal is to minimize the response time of applications, which is determined by process delay. Binwei et al. [2] suggested a framework to deal with an energy and latency aware workload assignment in distributed edge cloud computing. The communication delay and error rates are taken as constraints, and the objective is to minimize the communication of applications during offloading and to improve the battery life of devices. Xiang et al. [3] formulated a single latency aware workload assignment in the cloudlet network, where the communication delay and the process delay are taken as constraints and the average response time of the application is the objective of the study. Qiang et al. [4] and Shanhe et al. [5] formulated latency aware workload problems for computing-intensive real-time workloads and healthcare applications in the distributed edge network. These studies tried to reduce the end-to-end latency of applications via offloading and resource allocation methods. Jungmin et al. [6] and Amit Samanta et al. [7] proposed latency optimal workload assignment greedy strategies to optimize the objective function of applications. The objective function was to communicate delay and process delay during workload allocation in the distributed edge cloud network. Huy Trinh et al. [8,9] devised solutions based on dynamic programming based iterative greedy solutions to solve an energy-efficient and low-latency optimal workload in distributed edge cloudlet networks. These studies optimize the objective function of the problem, that is, the energy and latency, and obtained near-optimal solutions in the iterative process.

A.Rasheed et al., Lixing et al. and Dileep et al. [10–12] proposed dynamic programming-based iterative and full approximation methods based on Lyapunov optimization heuristics to solve the workload assignment problem in the distributed edge cloud network. The concave and convex optimization-based solution suggested solving the travelling salesmen's problem for workload in the network. All studies considered the energy and latency objectives during problem formulation and met the applications' Quality of Service requirements. Li et al. [13–15] and Ying Wah et al. [16] suggested energy, latency and cost-aware workload assignments in the distributed mobile edge/fog/cloudlet based cloud network. These studies solved the workload assignment based on NP-Hard scheduling heuristics and meta-heuristics. The goal was to minimize the makespan of applications. The considered problem is dynamic, and a Queue based solution is suggested to estimate the required resources for offloaded workload in the system. The communication delay, process delay and energy cost were taken as constraints of the study.

Aburukba et al. and Lakhan et al. [17] suggested mobility-aware workload assignment solutions for the distributed network. The processing delay, communication delay and progation delay are also considered during problem formulation and, based on these delays, the studies proposed solutions. The studies [19] suggested dynamic environment workload assignment aware strategies based on Genetic Algorithms and ant-colony meta-heuristics. The solutions are fully approximated, and exponentially gained optimal objectives in an exploration and exploitation environment. The studies [20–23] formulated dynamic content and failure-aware workload assignment-based scheduling and resource allocation techniques. Different heuristics, such as Hill Climbing, Earliest Deadline First, and Earliest Finish Time methods, suggested minimizing the tardiness latency of applications. Wang et al. and Lakhan et al. [24–27] introduced reinforcement learning-based resource allocation techniques and workload assignment strategies in the dynamic environment, where the network contents and load balancing situation of the computing nodes are considered to be constraints. The application-level latencies, such as average response time, were booted via a fully offloading technique in the Queue-based cloudlet cloud network. Li et al. [28] proposed mobility and fault aware methods in a dynamic environment where the latency and makespan of applications were considered as objective functions. The communication delay and the process delay were the constraints, and each workload had a deadline constraint.

To the best of our knowledge, the multi-layer latency-aware workload assignment in the cloudlet-based cloud network and the Q-learning migration workload have not yet been studied for mobile cloudlet based cloud applications. This study considered the QoS of applications and the multi-layer latency of the workload (e.g., round-trip delay, process delay and migration delay) with the roaming situation of E-Transports. This study suggested dynamic mixture solutions based on meta-heuristics and convex optimization to obtain the optimal application solution in dynamic and adoptive cloudlet-based cloud environments. In this way, the study optimizes the average response time of applications and runs them under QoS requirements.

3. Problem Description

This study devises a novel system, the Multi-Layer Latency Aware Workload Assignment of E-Transport IoT Applications in Mobile Sensors Cloudlet Cloud Networks, as shown in Figure 1. The primary objective is to boost the performances of E-Transport applications in distributed sensors cloudlet cloud networks. The system consists of different Internet of Things (IoT) E-Transport applications. At the same time, local mobile sensors decide where to send the workload for further execution. The computing nodes are local mobile sensors, cloudlets and cloud computing, which are geographically distributed and are connected to the different communication technologies. The software-defined network control plane is where communication nodes connect to each other to provide an efficient routing path to transport applications. This study considered the multi-latency aware workload assignment in the cloudlet-based cloud network. The considered workload is coarse-grained, where the thin mobile clients offload their workloads to the cloudlet based cloud for further processing. This study suggests using the Multi-Layer Latency Aware Workload Assignment (MLAWAS) framework, which consists of different global searching and local searching iterative methods and Q-Learning heuristics.

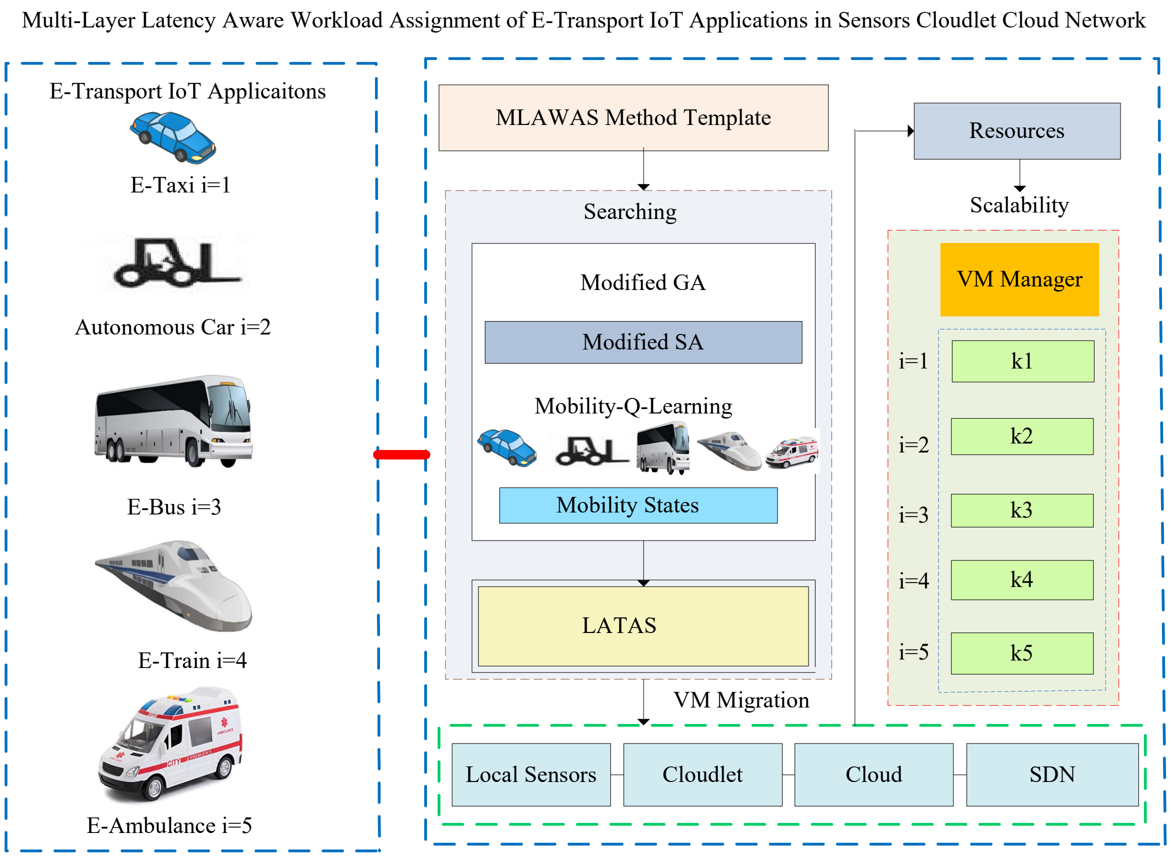

Figure 1. IoT E-Transport System.

3.1. Application Workload Characterization

In this work, all workloads are full offloading coarse-grained, consisting of different types of tasks. The Poisson process follows the arrival of the workload to the MCBC. The M/M/S-based queue in the system immediately allocates these workloads to the available optimal cloudlets for further execution. All workloads have QoS requirements (e.g., maximum latency tolerance).

3.2. Sensors Cloudlet Cloud Assignment Mechanism

This study formulates multi-latency aware workload assignment problems in cloudlet-based cloud networks where the objective is to minimize the average response of the workloads. This study proposed an offloading and execution system based on MLAWAS in a cloudlet-based cloud network as shown in Figure 2. There are three main layers in the system. The first layer is a mobile client where all applications are installed; the cloudlet client for offloading is asked if the total average response is optimal for offloading. The second layer is the cloudlet host, which consists of many homogenous servers. The client host consists of different virtual machine (VM) host components, which wrapped all coarse-grained applications into a single VM. The VM manager allocates the single computing node to a workload as a guest VM for execution. The cloudlet meta-data and service discovery are two components that aim to save application data and monitor the remaining resources of the cloudlet host during execution. The third layer is the remote cloud, which only executes the migrated workload of the application from the cloudlet host. The remote cloud also has meta-data and a discovery service, which saves the completed data and monitors cloud computing resources during the execution of the workload in the cloud computing. MCBC is a distributed network in which mobile devices are treated as thin clients, and cloudlet based clouds are thick computing nodes for processing the requested workload to their QoS requirements.

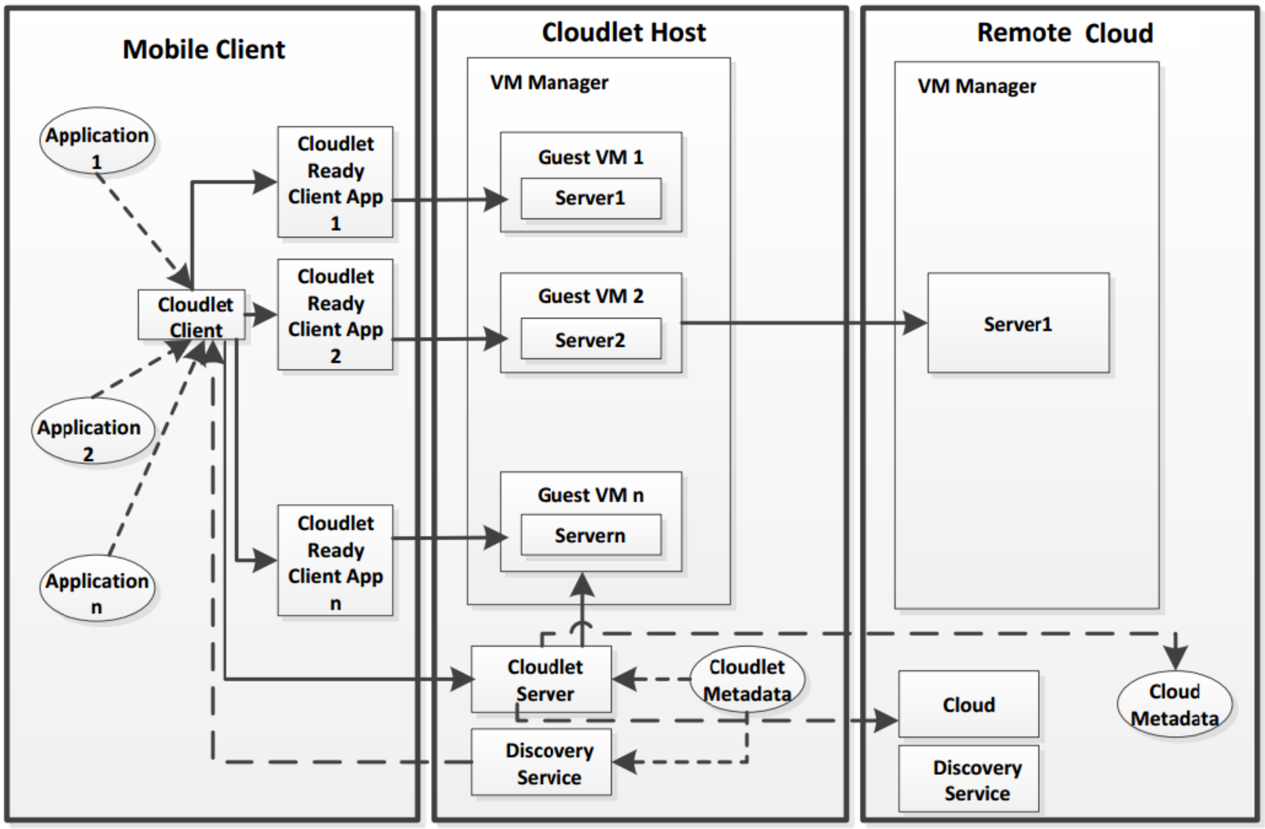

Figure 2. Proposed Multi-Latency Aware Offloading System Based on MLAWAS Framework.

3.3. System Model

This study considered the fine-grained workload of applications. The computing environment is geographically distributed in the network as shown in Figure 3, and it is similar to existing work [4]. There are three main aspects of the considered work: mobile E-Transports, cloudlet, and cloud computing. All the cloudlets are placed flexibly and are connected to the different base stations in the network. At the same time, all base stations are managed by a Software Defined Network (SDN) to provide an efficient routing path to the E-Transports during offloading in the network. The study also considered cloud computing services that execute the migrated workload of applications from cloudlet servers to avoid overloading. Each cloudlet is made up of heterogeneous servers where they have different computing speeds and resources. Multi-latency-aware aspects are considered in this study—communication latency (round-trip delay and migration) and process delay.

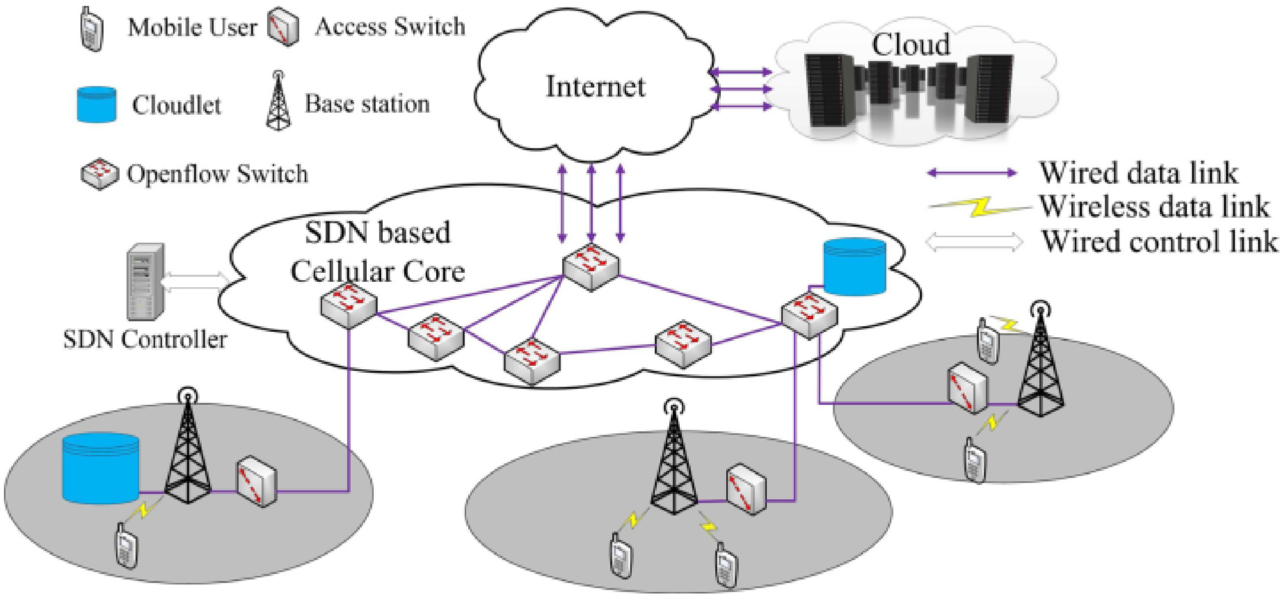

Figure 3. SDN Based Mobile Sensors Cloudlet Cloud Network Scenario.

3.4. Multi-Latency Based Problem Formulation

We examine the MCBC as the amalgam of a set of base stations (BS) and a set of mobile E-Transports under the coverage base station, where each base station is outfitted with one cloudlet as shown in Figure 3. As we already explained, the cloudlet is an extended small data center for computing and storage abilities. Every individual cloudlet has a set of similar types of virtual machines. Mobile E-Transports can execute and offload the task to the closest cloudlet, and uploading and downloading the data from the cloudlet are identical. The arrival rate of the mobile E-Transport i application workload requests follows a Poisson process, represented by λi, whereas $i \in I$. We describe c_i as the number of computation tasks generated by mobile E-Transport i, whereas d_i symbolizes the quantity of data to be downloaded by mobile E-Transport i. Let ϵ_k be the range of the cloudlet k for data processing and storage. The average service rate of the mobile device and the cloudlet are defined as θ_i and μ_k. We mock-up each base station B and mobile E-Transport i inside the coverage area of the base station (BS) as an M/M/1 queuing system. Normally, MU sends data requests, and then either the cloudlet or the remote cloud provides the results. This procedure engages the transmission between MU and the BS and the BS to the remote cloudlet. In simple terms, we define the e_{UM} transmission delay between MU and BS, whereas e_{MC} is the transmission delay between the base station and a remote cloud.

3.5. Process Delay and Network Delay

Here, we are using binary variables to show the vector dimension of all delays such as $X = \{x_{ik} : k \in K, i \in I\}$ with x_{ik} = $\{0, 1\}$ shows whether the mobile E-Transport accesses the associated cloudlet or not, that is,

$$x_{ik} = \begin{cases} 1, & \text{if } x_{xk} = 1 \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

In Equation (1), vector $Y = \{x_{ik} : k \in K, i \in I\}$ shows whether the computational task to offload is generated by the mobile E-Transport i to the cloudlet k or not, likewise,

$$y_{ik} = \begin{cases} 1, & \text{if } x_{xk} = 1 \ \text{ offload to cloudlet } k \text{ or cloud } k \\ 0, & \text{execute task locally at mobile device } k \end{cases} \tag{2}$$

Equation (2) determines the offloading of the application workload either to the cloudlet or the cloud.

3.6. Problem Formulation

We use the queuing model M/M/1-PS [19] to calculate the processing delay of any computing node k as follows:

$$C_k^i = \frac{y_{ik} c_i \lambda_i}{\mu_k - \sum_{i=1}^{I} y_{ik} c_i \lambda_i}. \tag{3}$$

Equation (3) determines the execution on the cloudlet, where $y_{ik} c_i \lambda_i$ is the workload computation task offloaded to cloudlet k, and $\frac{1}{\mu_k - \sum_{i=1}^{I} y_{ik} c_i \lambda_i}$ is the average execution delay of the mobile E-Transport task i at cloudlet server k with $\mu_k - \sum_{i=1}^{I} y_{ik} c_i \lambda_i > 0$. In the same way, the computation delay on computing node k occurs in the following way:

$$C_i^U = \frac{\left(1 - \sum_{k \in B}\right) c_i \lambda_i}{\theta_i - \left(1 - \sum_{k \in B}\right) \lambda_i}. \tag{4}$$

Equation (4) determines the computation delay at the mobile device, where $\left(1 - \sum_{k \in B}\right) c_i \lambda_i$ is the computation task amount that is locally executed at the mobile E-Transport device and it must be $\left(1 - \sum_{k \in B}\right) c_i \lambda_i > 0$.

If the request for data access to the cloudlet server is k, then the transmission delay between the mobile E-Transport and the cloudlet is as follows:

$$T_i^K = \sum_{k \in K} e_{UM}(y_{ik} c_i + d_i). \tag{5}$$

Equation (5) determines the communication delay between E-Transport and cloudlet. $y_{ik} c_i$ shows the volume of data at cloudlet k requested by the mobile E-Transports.

Additionally, the delay between the cloudlet and the remote cloud for the additional process is expressed as follows:

$$T_i^{K_C} = \sum_{k \in K} e_{MC}(1 - x_{ik}) d_i. \tag{6}$$

Equation (6) calculates the communication delay between the cloudlet and the cloud computing. $(1 - x_{ik})$ =0 shows that the requested amount of content for the processing is stored in the cloudlet k and the mobile E-Transport does not need to access the remote cloud.

Finally, the total Service Delay of all mobile E-Transports is the amalgam of the transmission delay and execution delay expression as follows:

$$\tau = \sum_{i \in I} \sum_{k=1}^{K} \left(C_i^K + C_i^U + T_i^{U_K} + T_i^{K_C} \right). \tag{7}$$

Consequently, we know it is an optimization problem and can be formulated as an optimization problem, which minimizes the τ with various constraints as shown in Equation (7).

$$minimize \quad \tau. \tag{8}$$

Equation (8) determines the objective function of the study.

$$s.t. \ \sum_{i=1}^{I} d_i x_{ik} \leq \epsilon_k, \ \forall k \in K, \ \forall i \in I. \tag{9}$$

Equation (9) shows that the requested workload of all applications must be less than the capacity of any computing k.

$$\mu_k - \sum_{i=1}^{I} y_{ik} \lambda_i > 0, \ \forall k \in K, \ \forall i \in I. \tag{10}$$

Equation (10) shows that all computing nodes easily handle overloading and overhead situations before offloading and workload assignment in the network.

$$\theta_i - \left(1 - \sum_{k=1}^{K} y_{ik} \right) \lambda_i > 0, \ \forall k \in K, \ \forall i \in I. \tag{11}$$

All computing nodes, such as mobile computing, cloudlet node and cloud node, must be stable in the network as defined in Equation (11).

$$x_{ik} = \{0, 1\}, \ \forall k \in K, \ \forall i \in I. \tag{12}$$

Each computing node can execute one workload at a time as defined in Equation (12).

$$\sum_{k \in K} x_{ik} = 1, \ \forall i \in I. \tag{13}$$

Each workload can be allocated to any single computing node as defined in Equation (13).

$$\sum_{i \in I} x_{ik} = 1, \ \forall k \in K. \tag{14}$$

The respective Equations (14) and (15) determine the binary variable, which shows that either the workload is assigned to any computing node during offloading and workload assignment in the network.

$$y_{ik} = \{0, 1\}, \ \forall k \in K, \ \forall i \in I. \tag{15}$$

All equations are numeric, which shows that the considered problem is a dynamic integer problem.

4. Proposed MLAWAS Algorithm Framework

This study considered the multi-latency aware workload assignment in a cloudlet-based cloud network. The considered workload is coarse-grained, where thin mobile clients offload their workloads to the cloudlet based cloud for further processing. This study suggested the Multi-Layer Latency Aware Workload Assignment (MLAWAS) framework, consisting of different global searching and local searching iterative methods, and Q-Learning heuristics. We defined the entire process of workload assignment based on MLAWAS components, as shown in Algorithm 1.

Algorithm 1: MLAWAS Algorithm

```
Input  :
        $\tau = \sum_{i \in I} \left( C_i^K + C_i^U + T_i^{U_K} + T_i^{K_C} \right)$;
        cloudlet service rate $\mu_k$;
        Mobile device $\theta_i$;
Output: Workload assignment of the mobile to optimal cloudlet
        $\mathbf{S} = \{x_{ik} \mid i \in I,\ k \in K\}$;
1  begin
2      foreach ($i$ to $I$) do
3          Call Modified Genetic Algorithm for Initial Workload Assignment;
4          $i \leftarrow k$ based on equation (7);
5          Improve the global search call Simulated Annealing Algorithm method;
6          $\tau \leftarrow \tau^*$;
7          Call LATAS method to improve the solution within local space;
8          Call VM migration method to move a workload from cloudlet to cloud;
9          Swapping Workload $i \leftarrow$ from node $k1$ to $k2$ ;
10         The migration will be done based on VM migration scheme;
11 End MLAWAS Framework
```

We defined the respective techniques of MLAWAS in the following subsections.

4.1. Improved Genetic Algorithm for Initial Workload Assignment

A mixture evolutionary optimizer algorithm was developed for the Service Delay. The proposed improved GA (Genetic Algorithm) and SA (Simulated Annealing) are merged to provide the optimal geo-distributed cloudlet network. The proposed algorithm is proficient at defeating the untimely convergence of the Genetic Algorithm and helped the escape from the optimal local solution. We have proposed the modified classical Genetic Algorithm to resolve the optimization problem efficiently.

The above-given problem is not easy and is non-linear for the reason that x_{ik} and y_{ik} both have different directions. We implement our problem in a modified Genetic Algorithm and a Simulated Annealing algorithm, compared with classical Simulated Annealing and genetic heuristics. We compare individual components with classical heuristics and find some improvement as follows:

1. In the Genetic Algorithm (GA) cross-process, we replicate the Simulated Annealing (SA) operation for incoming new individuals that can avoid heuristics converging to the optimum local value with some low fitness individuals passing onto the next generation.
2. The fitness function calculated the incoming individual fitness value after the crossover operation and the mutation operation, correspondingly. This way, our techniques save the best individuals from being lost during the process of evolution.

It is worth remembering that the individuals are promising solutions to the given problem that is to be optimized.

Fundamentally, as articulated in Algorithm 1, given the solution n_s = $\{x_{ik}y_{ik}\}$ where $k \in K$ and $i \in I$ are the 2D-dimensional vectors that can be filled with either 1 or 0, where the first D represents the value of x_{ik} and the second D represents the value of y_{ik}, respectively. The intention of the feasible solution given in the proposed heuristics was to guarantee all constraints. Furthermore, the solution of s with the given fitness function is $f(s)$ and the probability that every individual would be inherited by the incoming next generation is $P(s)$ as follows:

$$p(s) = \frac{f(s)}{\sum_{s=1}^{2D} f(s)}. \tag{16}$$

Equation (16) shows the probability of the function value. There is another type of cumulative probability as follows:

$$Q(s) = \sum_{i=1}^{s} P(i). \tag{17}$$

Equation (17) shows the commutative probability of the objective function during the initial assignment of workloads.

In Algorithm 2, the original population updated by steps 8–12, Pop, selects the better individual with respect to $Q(k)$. Steps 15–21 show the crossover procedure from a solution; it randomly selects a figure of bit positions and then replaces the two bits in the same position as two individuals. Step 22 provides the update value of Pop_1 to support Algorithm 3. In Algorithm 3, we measure all the individuals that belong to two dissimilar Pop_1 and Pop_2 populations. There exists the probability that bad individuals will be inherited by the subsequent generation with a certain probabilistic expression such as $\exp(-(f(n_s') - f(n_s))/T))$. Steps 16 and 30 provide the surety that the crossover procedure and the mutation solution strictly satisfy all constraints in their given equations. Steps 24–26 describe the comparison between Pop_1 and Pop_2 and record the lowest fitness values of both of them. Steps 28–35 express the procedure of the mutation, while step 32 initializes or generates the new optimal solution ni_c and turns over $(0 \leftarrow 1 \; or \; 1 \leftarrow 0)$. Furthermore, Steps 38–40 randomly recorded the lowest fitness values and compared them with the fitness values in steps 37 and 25, which is the optimal solution that we observed and found.

Algorithm 4 always returns the optimal solution through the global search space, whenever a Genetic Algorithm returns a feasible solution locally; then Simulated Annealing helps it to escape from the local optima.

4.2. E-Transport Applications Mobility and Migration Latency

The E-Transport $i \in I$ roams in the geographically distributed cloudlets $k \in K$ through the cellular network from base station to base station along with the VM service that is in progress, as shown in Figure 4. It needs to be migrated to the E-Transport, but this process is stochastic and network behaviors periodically change; this leads to many challenges associated with propagation latency, such as congested nodes, high traffic and lower bandwidth. Our goal is to migrate the running VM and transport application with lower propagation delay without interruptions and disruptions to the VM service. Our method always finds the optimal shortest path among all environment spaces. Here, we introduce the utility table of each cloudlet, which saves the data of the current E-Transport and the data to be migrated via the VM migration method, and we save another utility table of the cloudlet server as shown in Figure 4. The E-Transport mobility scenario will be discussed in the following way.

E-Transport Applications Mobility Scenario

All E-Transport applications can move from one place to another place, for example, during a business trip or roaming between different locations. The flexible cloudlet-based cloud is geographically distributed in the network, and all computing nodes are connected via base stations. If the application $i = 1$ migrates the transport from one node k = 1 to another node k = 2 and the utility table that transfers the workload runs the information to another cloudlet for further execution, the main advantage of the mobility and migration of the workload is avoiding overloading and overhead. Figure 4 shows the scenario of

E-Transports roaming between different base stations, and all cloudlet and cloud services are ubiquitous and omnipresent.

Algorithm 2: Improved Genetic Annealing Algorithm

```
Input :
        size R of the population;
        Probability $P_o$ of the crossover;
        the $P_1$ is the probability of the mutation;
        Number of iteration is $N$
Output: $\tau_{min}$
begin
    initialization;
    $n_s$ is the randomly generated of feasible solutions $K$ ;
    calculate the $f(n_s)$ according to step-8;
    save all the $n_s$ and $f(n_s)$ in pop;
    $T_{min}$ = min$f(n_s)$,$n_s \in pop$;
    for ($k$ = 1 to $K$) do
        for ($i$ = 1 $to$ $R$) do
            step-15,16 calculate $P(i)$ and $Q(i)$;
            if $Q(i) > rand$ then
                select $i$ and save it in $Pop_1$
        //crossover with probability $P_0$ in $Pop_1$ ;
        for ($i = 1$ $to$ $K$) do
            while ($n'_a$ and $n'_b$ all are feasible) do
                randomly select two solutions $n_a$ and $n_b$ from $Pop_1$;
                generate two new feasible solutions $n'_a$ and $n'_b$ by one point-crossover;
            calculate $(n'_a)$ and $f(n'_b)$ and save them in $Pop_2$;
        update $Pop_1$ based on Algorithm 2;
        $\tau'_{min}$ = min$f(n_s)$,$n_s \in Pop_1$;
        if ($\tau_{min} > \tau'_{min}$) then
            update $\tau_{min}$ with $\tau'_{min}$;
        The mutation with probability $P_1$ in $Pop_1$;
        for ($i = 1$ $to$ $k$) do
            randomly select a solution $n_c$ from $Pop_1$ while ($n_c$) do
                randomly select mutation position in $n_c$;
                generate a new feasible solution $n'_c$;
                update $n_c$ with $n'_c$ in $Pop_1$;
        calculate $f(n_s)$,$n_s \in Pop_1$;
        $\tau'_{min}$ = min$f(n_s)$,$n_s \in Pop_1$;
        if ($\tau_{min} > \tau'_{min}$) then
            update $\tau_{min}$ swap $\tau'_{min}$;
        End Loop
return $\tau_{min}$;
```

We model the propagation latency as follows:

1. The set of network and agent state cloudlets $k \in K$.
2. The set of VM migration actions $a \in A$ of the agent.
3. $P_a(k,k') = P_r(k_{t+1} = k' \mid k_t = k, a_t = a)$ describes the probability of state k to state k' transition in action a.
4. $R_a(k,k')$ describes the instant reward following k to k' transition in action a.
5. All kinds of rules that explain what the agent cloudlet Controller observes.

All rules are stochastic because the behaviour of distributed cloudlet networks randomly changes node to node. It depends upon the cases. Sometimes agents know to observe the current environment cloudlet $k \in K$ state; sometimes it is fully observed and sometimes partially observed. All sets of action $a \in A$ are stopped if it reaches zero. Furthermore, it is not reducible. Because of the stochastic nature of networks, learning agents in cloudlet environments follow discrete time t, under action a_t; they contain a set of reward values from a set of actions $a_t \in A$. The E-Transport moving from cloudlet state to cloudlet state can be expressed as $k_t + 1$ and reward factor $r_t + 1$ belongs to the transition process and is expressed as $(k_t, a_t, k_t + 1)$.

Algorithm 3: Improved Simulated Annealing

```
Input  :
        $f(n_s), n_s \in Pop_1$;
        $f(n'_s), n_s$ ;
        initial temperature $T$;
        rate $A$ of annealing;
        Number of $I$ iterations;
Output: $Pop_1$
1 begin
2     while ($T > 0$) do
3         for $i$ to $I$ do
4             if $f(n'_s) < f(n_s)$ then
5                 update $n_s$ with $n'_s$;
6                 if $\exp(-(f(n'_s) - f(n_s))/T)) >rand$ then
7                     update $n_s$ with $n'_s$;
8             $T$=$T * A$;
9     return $Pop_1$
```

Algorithm 4: LATAS Algorithm

```
Input  :
        $\tau = \sum_{i \in I} \left( C_i^K + C_i^U + T_i^{U_K} + T_i^{K_C} \right)$;
        cloudlet service rate $\mu_k$;
        Mobile device $\theta_i$;
Output: Workload Assignment of the mobile to Optimal cloudlet $\mathbf{S} = \{x_{ik} \mid i \in I,\ k \in K\}$;
1 begin
2     Initialize S = 0;
3     The request of all transport application workloads applications generated in descending order ;
4     $i^* = 1$;
5     while ($i^* \leq \mid I \mid$) do
6         Find the underutilized optimal cloudlet with lower delay for transport application $i^*$, based
            on equation ;
7         Set $x^*_{ik^*} = 1$;
8         $i^* = i^* + 1$;
9     return S
```

In our problem, the data model of the cloudlet environment is known. Still, we are looking for the optimal solution, so this is the planning problem and it must be converted into a machine learning specified problem.

The learning cloudlet agent requires an optimal searching mechanism to randomly select an action without degrading the QoS, for example, the latency and bandwidth. We solve our problem of the principle of optimality with respect to two factors which are as follows:

The action in the current cloudlet agent is replicated as a plan called policy π.

The cloudlet policy:

$$\begin{aligned} \pi :\ K \times A \rightarrow [0, 1] \\ \pi : (a \mid k) = P(a_t = a \mid k_t = k). \end{aligned} \tag{18}$$

In Equation (18), the policy π always provides the possibility in cloudlet k state while taking action a.

The cloudlet value function:

Value function $V_\pi(k)$ is expressed as the preliminary predictable return with cloudlet state k and policy π; the cloudlet value function approximates how high the quality is in the given cloudlet state.

$$V_\pi(k) = E[R] = E\left[\sum_{t=0}^{\infty} \gamma^t r_t \mid k_0 = k \right]. \tag{19}$$

Equation (19) measures the value function, whereas the randomly generated variable R shows the return and is expressed as the sum of the value of discount rewards; in $R = \left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$, value t_t shows a reward at step t and $\gamma \in [0,1]$ expresses the discount rate.

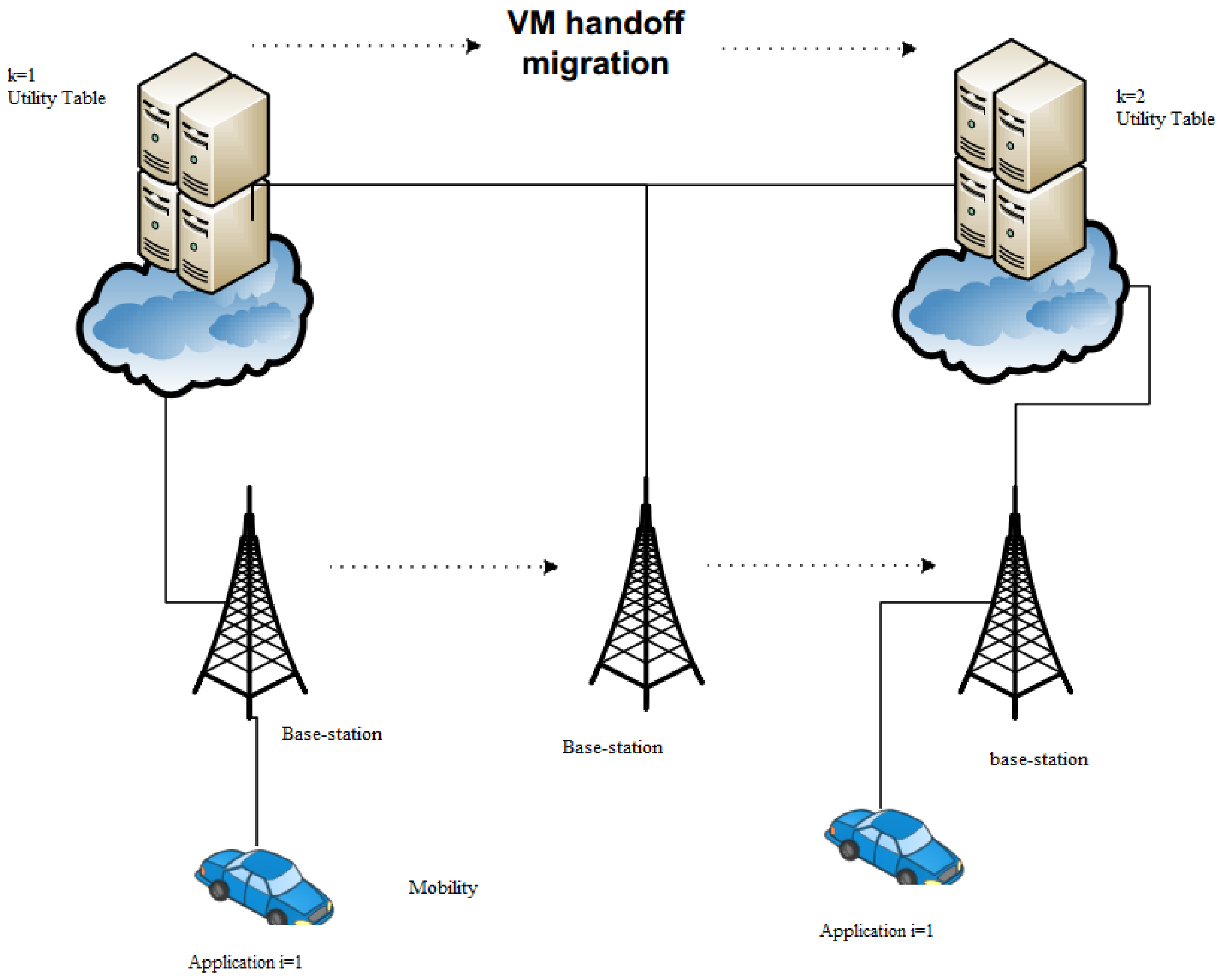

Figure 4. E-Transport Mobility in the Network.

Algorithm 5 determines workload migration from one computing node to another computing node based on the Q-Learning model. States show that a new optimal objective function must be the best as compared to the current solution.

Figure 5 shows that the trained model in Q-Learning performs well, and improves and chooses the optimal computing node when the workload migrates between computing nodes.

Algorithm 5: E-Transport Mobility and Migration latency.

```
Input  :
        Cloudlet States $K = \{k_1, k_2, \ldots n_k\}$;
        a set of Actions $A = \{a_1, a_2, \ldots a_n\}$ $A : K \Rightarrow A$;
        Reward Function $R : K \times A \to R$;
        Probability transition function $T : S \times A \to S$;
        Learning rate $\alpha \in [0,1]$;
        Discount factor $\gamma \in [0,1]$;
        Procedure Q-Learning $(S, A, R, T, \alpha, \gamma)$;
Output: Max $Q$;
1  begin
2      initialize $Q := S \times A \to R$;
3      while ($Q$ isn't converged) do
4          Migrate VM from state $k \in K$;
5          while ($k$ isn't terminal) do
6              Calculate $\pi$ regard to Q and exploration strategy $\pi((k)\ \mathrm{arg}Max_a$ Q(k,a));
7              $a \leftarrow \pi(k)$;
8              $r \leftarrow R(k,a)$  Received the reward ;
9              $k' \leftarrow T(k,a)$  Received the new cloudlet state ;
10             $Q(k',a) \leftarrow (1-\alpha)\ Q(k,a) + \alpha(r + \alpha.Max_a\ Q(k',a'))$;
11     return $Q$
```

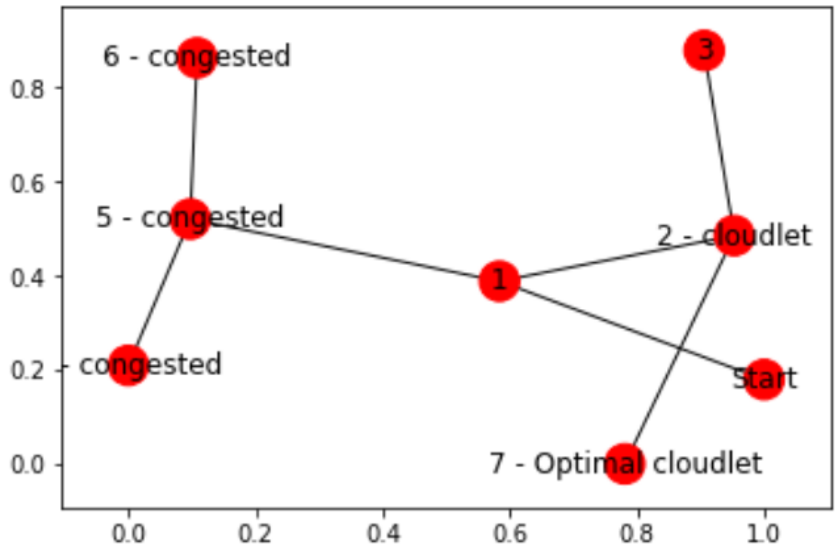

Figure 5. Training Model of Proposed Q-Learning.

5. Experimental Design

The performance evaluation shows the effectiveness of the proposed algorithm frameworks when they run a healthcare workload in the cloudlet-based cloud network. This paper implemented the proposed framework in the simulation form. The related parameters of the simulation are described in Table 2. The coarse-grained healthcare workload consists of different types of tasks such as text, video and images. This study implemented different baseline approaches and proposed schemes as defined in the following.

- Conventional Method: This method is implemented widely in different studies [1–8] to solve the offloading and scheduling problem for the coarse-grained E-Bus workload. The goal was to minimize execution costs and the communication of the workload during offloading and assignment in the cloud network. The method consisted of four main components: profiling of workload, workload assignment, migration, and ordering of tasks. The centralized cloud-based network implemented in these studies solves the delayed optimal workload of applications.

- Lean Algorithm: The Latency Effective Aware Algorithm is a framework which allocates workload to different available cloudlets. The framework consists of the assignment part and the migration part of the applications. This framework has been implemented in different studies to solve the healthcare workload of applications [9–15].
- MLAWAS Framework: This proposed method is composes of different components, such as assignment, migration, local and global search, and is considered the dynamic environment in this study.

Table 2. System and Vehicular Mobility Simulation Parameters.

Simulation Parameters	Values
Windows OS	Linux Amazon GenyMotion
Centos 7 Runtime	X86-64-bit AMI
Languages	JAVA, XML, Python
Android Phone	Google Nexus 4, 7, and S
Experiment Repetition	160 times
Simulation Duration	12 h
Simulation Monitoring	Every 1 h
Evaluation Method	ANOVA Single and Multi-Factor
Amazon On Demand Service	EC2 t3
K	Number of cloudlets
Detail Resource Specification	Shown in Table 4
Mobility	Waypoint Model
Vehicular Model	VANET++

5.1. Simulation Parameters

First scenario (static): We examined the performance of the proposed MLAWAS algorithm in static traffic conditions, where vehicles do not move in this simulation scenario. The static traffic scenario allows us to demonstrate the proposed algorithm's fundamental characteristics. The maximum density and standard deviation (SD) were set to 55 cars in a 100 m radius and 1000 m, respectively, on a straight double lane road (5 km). In the second scenario, vehicle mobility was taken into account, and the algorithm E-Transport Mobility and Migration latency was used to handle both application mobility and workload migration.

The simulation was performed on local machines, where baseline classes were taken from edgecloudsim and VANET++ [14] vehicular environment integrated into edgecloudsim [13]. The goal was to exploit the the available abstract of edgecloudsim and VANET++ for the applications' cloudlet implementation and vehicular environment. For all the cloudlets K, in the simulation environment, we implemented E-Transport classes of edgecloudsim and tested on the genymotion [15] environment with the first scenario. In the second scenario, we named the VANET++ classes and implemented waypoint mobility classes to analyze the mobility and migration of the workload in the system. The simulation parameter in Table 2 shows that the system implemented in JAVA and XML allows communication among different nodes such as cloudlet, cloud and applications interfaces. All the applications, as shown in Table 3, are E-Transport types in various domains such as E-Bus, E-Ambulance, and E-Taxi. These applications covered the healthcare aspects and e-passengers applications, which were integrated into the simulation environment at the config of edgecloudsim during simulation. This study implemented coarse-grained workloads with different types of tasks as defined in Table 3.

Table 3. E-Transport Workload.

Workload	W_i (MB)	W_i' (MB)	Image Tasks	Video Tasks	Text Tasks	I
E-Bus-i_1	825	5.2	100	100	300	500
E-Train-i_2	631	6.3	150	200	350	700
E-Taxi-i_3	645	7.4	100	200	500	800
E-Ambulance-i_4	755	8.7	200	200	600	1000
E-Mobile-Healthcare-i_5	555	3.7	400	100	600	200

The cloudlet based cloud network was deployed to run the workload tasks in this study. The cloudlet based cloud network configuration is defined in Table 4.

Table 4. Mobile Cloudlet Based Cloud Network.

Resources	**Cloudlet**			**Public Cloud**
	k_1	k_2	k_3	k_4
ζ_k	Small 2 V CPU	Medium 4 V CPU	Large 8 V CPU	Extra Large 24 V CPU
$CORE$	1	1	2	4
ϵ_k	500 GB	1000 GB	1500 GB	3000 GB
$Run-Time$	X86	X86	X86	X86

Table 4 defines the resource specification of the computing nodes with their characteristics and features.

5.2. System Implementation and Mobility Aware Dataset

This study designed transport applications on the specific interface, for example, the GenyMotion emulator of the android phone, which is then the directly connected base station. These base stations are geographically distributed and connected with distributed cloudlets and are controlled by the SDN controller. The study implemented a real-time dataset https://everywarelab.di.unimi.it/lbs-datasim (accessed on 18 May 2021), which is implemented in this work [29] on mobility for E-Transport movements in the network.

The study implemented real-time E-Transport mobility dataset-based simulation libraries, as defined in Table 5, in which the distance between the two base stations was about 1 KM. Thirty base-stations were considered with three cloudlet networks. Furthermore, 100,000 E-Transport users were considered during the simulation in this study. Figure 6 shows the map of the road of Milan where this dataset was implemented in practice, and we used this open-source data for the mobility features of the study.

Table 5. Mobility Dataset.

Road Network	**Simulation-Area**	**Map-Resolution**	**Number of Base-Stations**	**Number of Cloudlets**
Milan road network	17×17	50	3	4

5.3. Data Performance Method

The study tested four benchmark mobile cloud applications; their specifications are presented in Table 3. We tested applications that generate data (i.e., config file obtained data via profiling technologies and task scheduling heuristics) from different applications via analysis of variance (ANOVA). At the same time, ANOVA is an efficient parametric technique for examining the algorithm-generated data of mobile cloud applications from experiments. We exploited *t*-tests and dependent and independent random variables using the one way ANOVA method in order to assess the proposed method's efficiency.

Figure 6. Road network of Milan during Mobility.

5.4. Result Discussion

This section presents the evaluation performances of different methods to solve the workload assignment problem of applications. The results are compared via different methods and the performances of the workloads are compared with reference to their constraints during execution. The component performances are presented in their different subsections.

5.4.1. Communication Delay

Initially, the study considered the full offloading technique to migrate the workload from E-Transport devices to a cloudlet-based cloud network for execution. Due to the implementation of cloudlet and cloud in different networks, there is a communication delay during the offloading of the workload to the computing nodes. MLAWAS allows the workload when there is a shorter communication delay in the cloudlet or the cloud for execution. LEAN and conventional methods also focused on the communication delay during offloading; however, they did not focus on the delay between the wireless network and computing nodes, such as the cloudlet and cloud network—in this way, the communication increases in the different time intervals. Figure 7 shows that MLAWAS gained a shorter communication delay during the offloading of the workload as compared to the LEAN and the conventional method.

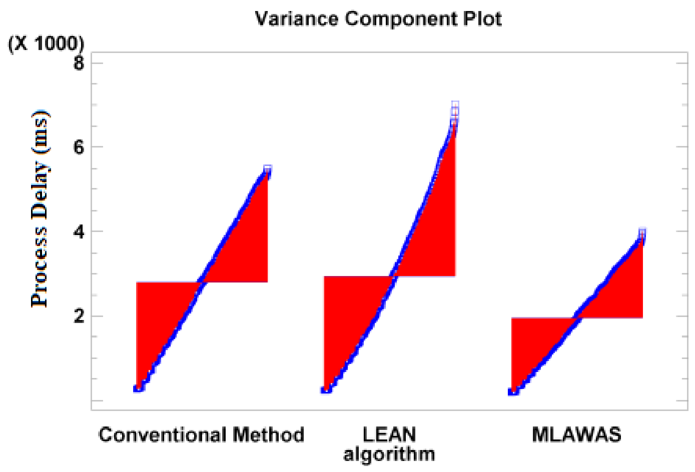

Figure 7. Communication Delay of Workload During Offloading.

The round-trip time (RTT) is shown in seconds during offloading and execution in the cloudlet-based cloud network. The E-Transport determines the round-trip from the wireless to the cloudlet or cloud computing nodes. Figure 8 shows that MLAWAS gained a smaller round-trip delay during the offloading and execution of the workload in the network.

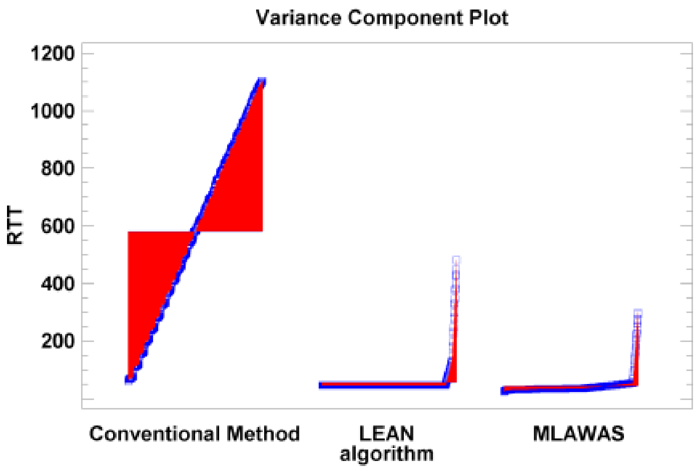

Figure 8. Round-Trip Time (RTT) of Workload During Offloading.

There is a migration delay between E-Transport and cloudlet, and cloudlet and cloud computing during the offloading and execution processes of the workload. This controls the load balancing situation. When the cloudlet becomes overloaded, it will migrate some workload to the cloud for further execution. However, LEAN and conventional methods did not consider the migration delay in their studies. Figure 9 shows that MLAWAS gained a shorter migration delay as compared to existing baseline methods during the migration of the workload from cloudlet to cloud for further processing.

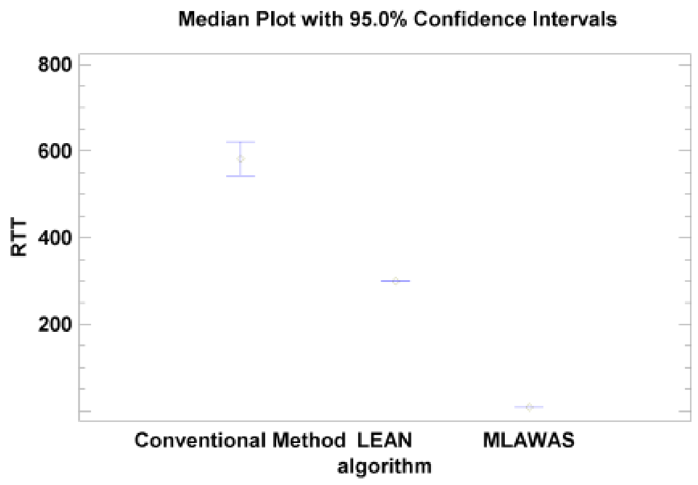

Figure 9. Migration Delay of Workload from Cloudlet to Cloud.

There is a migration delay between E-Transport and cloudlet, and between cloudlet and cloud computing during the offloading and execution processes of the workload. In these two ways, we migrate the workload in the study. Figure 9 shows the performance of workload migration from cloudlet to cloud. However, the study also considered the virtual machine migration between cloudlet and cloud. Figure 10 shows that MLAWAS gained a shorter migration delay as compared to existing baseline methods during the migration of workload from cloudlet to cloud for further processing.

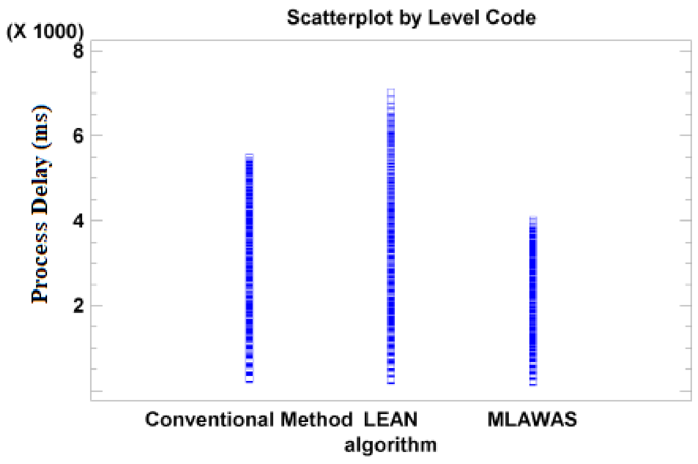

Figure 10. VM Migration Delay.

5.4.2. Processing Delay

The cloudlet delay means the computing delay of the workload at different computing nodes during execution. The cloudlet based cloud offers different computing nodes to run the workload of the applications. For instance, a cloudlet is a small data center consisting of different servers to run the application in proximity to E-Transports. It is a latency optimal data center where the application has small end-to-end latency. Due to the limited resource capability of the cloudlet servers, they can further migrate the workload to cloud computing for further execution. In this way, the load balancing situation and the overhead situation will be overcome during the processing of the workload. Figures 11 and 12 show that MLAWAS outperformed existing LEAN and conventional baseline approaches in terms of the execution of workload when considering load balancing and the overhead situation of nodes. However, existing approaches only focus on computing delay based on their computing speed rather than their overloading and overhead situation. Therefore, they had a longer delay as compared to MLAWAS during the execution of the workload.

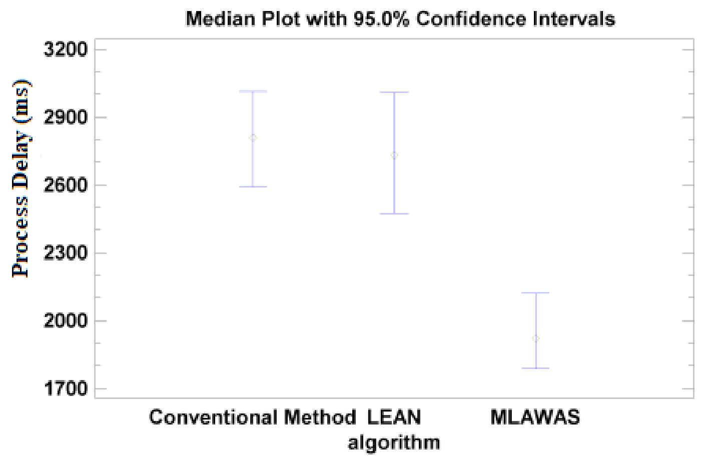

Figure 11. Processing Delay of Workload.

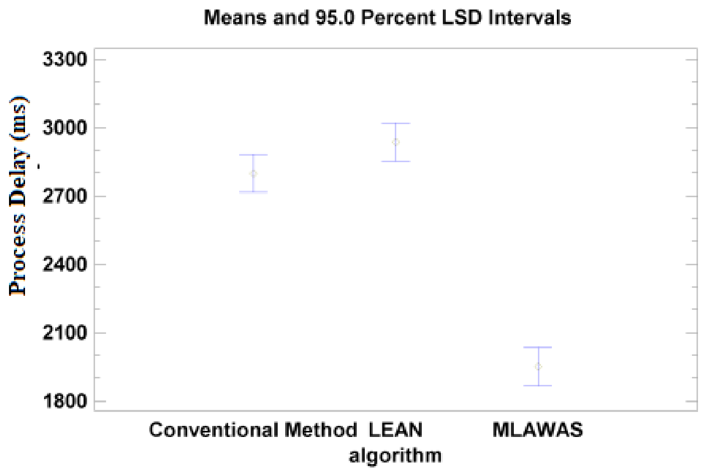

Figure 12. Processing Delay of Workload.

5.4.3. Average Response Time

The average response time of applications during offloading and execution is the study's objective and that of existing studies. The average response time (ms) combines communication time and execution time during the offloading and execution of the workload in the cloudlet-based cloud network. Communication delay takes different forms such as round-trip delay, migration delay, and offloading in the network.

Figure 13 shows the performances of the average response time of the workload in the cloudlet-based cloud network. This performance indicates that MLAWAS outperforms existing LEAN and conventional baseline approaches in terms of the average response time of applications. The main reason is that MLAWAS adopts dynamic changes in the network whereas MLAWAS exploits the Q-Learning approach compared to the greedy method, which is implemented by the existing method. The limitation of greedy and heuristic methods is that they can verify the optimal performance of applications at the end or initial stages of the workload assignment. However, Q-Learning in MLAWAS calculates the performance of the application at each step during offloading and execution. That is why MLAWAS performs better than the greedy methods, which are implemented in the existing method during offloading and scheduling.

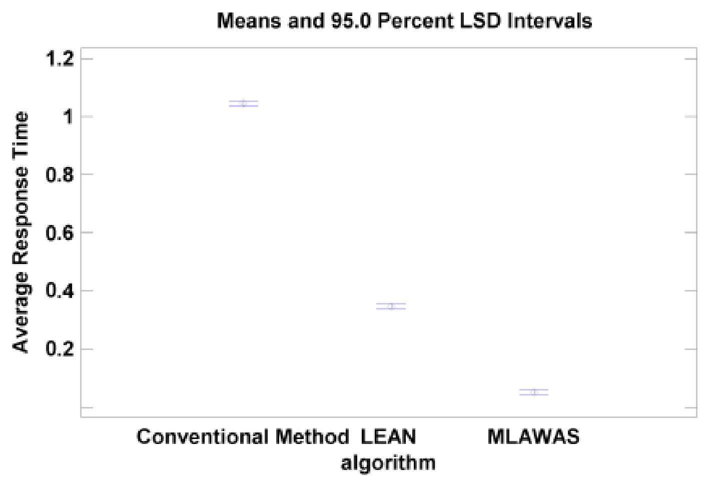

Figure 13. Average Response Time of Workload Without Round-Trip and Migration Delay.

Figure 14 shows the optimal performance of the average response time of applications. The main reason is that MLAWAS adopts dynamic changes in the network whereas MLAWAS exploits the iterative approach compared to the greedy method, which is implemented by the existing method. The limitation of greedy and heuristic methods is that they can verify the optimal performance of applications at the end or initial stages of the workload assignment. However, the iterative approach in MLAWAS calculates the application's performance at each step during offloading and execution. That is why MLAWAS performs better than the greedy methods, which are implemented in the existing method during offloading and scheduling.

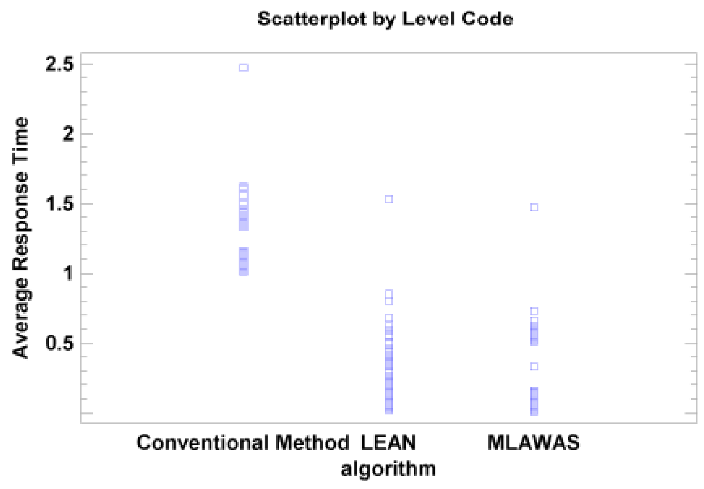

Figure 14. Average Response Time of Workload Without Migration Delay.

Figure 15 shows the optimal performance of the average response time of applications. The main reason is that MLAWAS adopts dynamic changes in the network where MLAWAS exploits the iterative approach as compared to the greedy method, which is implemented by the existing method. The limitation of greedy and heuristic methods is that they can verify the optimal performance of applications at the end or initial stages of the assignment of the workload. However, the iterative approach in MLAWAS calculates the performance of the application at each step during offloading and execution. That is why MLAWAS performs better as compared to the greedy methods, which are implemented in the existing method during offloading and scheduling.

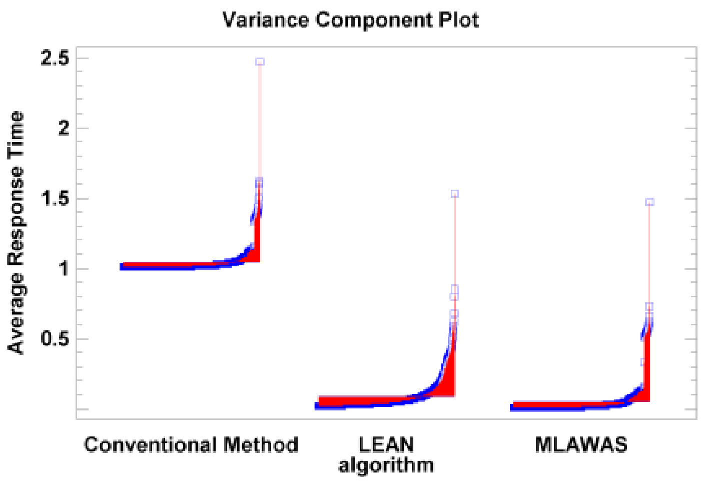

Figure 15. Average Response Time of Workload with Round-Trip and Migration Delay.

6. Conclusions and Future Work

Mobile cloudlet based cloud computing is an emerging hybrid computing architecture in which the workload of applications is executed at different nodes to gain lower end-to-end latency. Nonetheless, many delays are present in cloudlet-based cloud networks, such as communication delay, round-trip delay, and migration delay during cloudlet-based cloud network workload. The distributed execution of workloads at different computing nodes during an assignment is a challenging task. This paper proposes a novel Multi-layer Latency (e.g., communication delay, round-trip delay, and migration delay) Aware Workload Assignment Strategy (MLAWAS) to allocate the application's workload into optimal cloudlets and clouds. MLAWAS consists of different components, such as the Q-Learning aware assignment and Iterative methods, which distribute workload in a dynamic environment where runtime changes of overloading and overheating remain controlled. The migration of workload and VM migration are also part of MLAWAS. The goal is to minimize the average response time of the applications. Simulation results demonstrate that MLAWAS earns the minimum average response time as compared with two other existing strategies. In the future, we will extend our work from the security perspective, where blockchain-enabled technology and mobility of the Internet of Things (IoT) will be considered more in the current network. The work will introduce a new framework based on the serverless model—instead of a virtual machine based on container technology—which is lightweight and cost-optimal.

Author Contributions: In the study, all authors have made an equal contribution to the work. All authors have read and agreed to the published version of the manuscript.

Funding: This work fully funded by Kristiania University College, Department of Technology, Mobile Technology Lab, Norway. This work is financially supported by the Research grant of PIFI 2020 (2020VBC0002), China.

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

Acknowledgments: This work fully developed at Research Lab of AI and Information Security Department of Computer Science and IT BBSUL, Karachi, Pakistan. This work fully funded by Kristiania University College, Department of Technology, Mobile Technology Lab, Norway. This work is financially supported by the Research grant of PIFI 2020 (2020VBC0002), China.

Conflicts of Interest: The authors declare that there is no conflict of interest.

References

1. Lakhan, A.; Memon, M.S.; 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**, 1–23. [CrossRef]
2. Lakhan, A.; Ahmad, M.; Bilal, M.; Jolfaei, A.; Mehmood, R.M. Mobility aware blockchain enabled offloading and scheduling in vehicular fog cloud computing. *IEEE Trans. Intell. Transp. Syst.* **2021**. [CrossRef]
3. Sun, X.; Ansari, N. Latency aware workload offloading in the cloudlet network. *IEEE Commun. Lett.* **2017**, *21*, 1481–1484. [CrossRef]
4. Lakhan, A.; Mastoi, Q.U.A.; Elhoseny, M.; Memon, M.S.; Mohammed, M.A. Deep neural network-based application partitioning and scheduling for hospitals and medical enterprises using IoT assisted mobile fog cloud. *Enterp. Inf. Syst.* **2021**, 1–23. [CrossRef]
5. 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] [PubMed]
6. Son, J.; Buyya, R. Latency-aware Virtualized Network Function provisioning for distributed edge clouds. *J. Syst. Softw.* **2019**, *152*, 24–31. [CrossRef]
7. 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.
8. Trinh, H.; Chemodanov, D.; Yao, S.; Lei, Q.; Zhang, B.; Gao, F.; Calyam, P.; Palaniappan, K. Energy-aware mobile edge computing for low-latency visual data processing. In Proceedings of the 2017 IEEE 5th International Conference on Future Internet of Things and Cloud (FiCloud), Prague, Czech Republic, 21–23 August 2017; pp. 128–133.
9. Trinh, H.; Calyam, P.; Chemodanov, D.; Yao, S.; Lei, Q.; Gao, F.; Palaniappan, K. Energy-aware mobile edge computing and routing for low-latency visual data processing. *IEEE Trans. Multimed.* **2018**, *20*, 2562–2577. [CrossRef]
10. Mahesar, A.R.; Lakhan, A.; Sajnani, D.K.; Jamali, I.A. Hybrid delay optimization and workload assignment in mobile edge cloud networks. *Open Access Libr. J.* **2018**, *5*, 1–12. [CrossRef]
11. Chen, S.; Jiao, L.; Liu, F.; Wang, L. EdgeDR: An Online Mechanism Design for Demand Response in Edge Clouds. *IEEE Trans. Parallel Distrib. Syst.* **2021**. [CrossRef]
12. Gao, B.; Zhou, Z.; Liu, F.; Xu, F.; Li, B. An online framework for joint network selection and service placement in mobile edge computing. *IEEE Trans. Mobile Comput.* **2021**. [CrossRef]
13. Sajnani, D.K.; Mahesar, A.R.; Lakhan, A.; Jamali, I.A. Latency Aware and Service Delay with Task Scheduling in Mobile Edge Computing. *Commun. Netw.* **2018**, *10*, 127. [CrossRef]
14. Lakhan, A.; Li, X. Transient fault aware application partitioning computational offloading algorithm in microservices based mobile cloudlet networks. *Computing* **2020**, *102*, 105–139. [CrossRef]
15. Lakhan, A.; Li, X. Content Aware Task Scheduling Framework for Mobile Workflow Applications in Heterogeneous Mobile-Edge-Cloud Paradigms: CATSA Framework. In Proceedings of the 2019 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom), Xiamen, China, 16–18 December 2019; pp. 242–249.
16. Chen, Q.; Zheng, Z.; Hu, C.; Wang, D.; Liu, F. On-edge multi-task transfer learning: Model and practice with data-driven task allocation. *IEEE Trans. Parallel Distrib. Syst.* **2019**, *31*, 1357–1371. [CrossRef]
17. Jin, P.; Fei, X.; Zhang, Q.; Liu, F.; Li, B. Latency-aware VNF chain deployment with efficient resource reuse at network edge. In Proceedings of the IEEE INFOCOM 2020-IEEE Conference on Computer Communications, Toronto, ON, Canada, 6–9 July 2020; pp. 267–276.
18. Khoso, F.H.; Lakhan, A.; Arain, A.A.; Soomro, M.A.; Nizamani, S.Z.; Kanwar, K. Finedge: A Microservice-Based System for Industrial Internet of Things in Fog-Cloud Assisted Network. *Eng. Technol. Appl. Sci. Res.* **2021**, *11*, 7029–7032. [CrossRef]
19. Lakhan, A.; Khan, F.A.; Abbasi, Q.H. Dynamic Content and Failure Aware Task Offloading in Heterogeneous Mobile Cloud Networks. In Proceedings of the 2019 International Conference on Advances in the Emerging Computing Technologies (AECT), Al Madinah Al Munawwarah, Saudi Arabia, 10 February 2020; pp. 1–6.
20. Wang, T.; Xu, H.; Liu, F. Multi-resource load balancing for virtual network functions. In Proceedings of the 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS), Atlanta, GA, USA, 5–8 June 2017; pp. 1322–1332. [CrossRef]
21. Feng, C.; Sun, M.; Zhang, J. Reinforced Deterministic and Probabilistic Load Forecasting via *Q*-Learning Dynamic Model Selection. *IEEE Trans. Smart Grid* **2019**, *11*, 1377–1386. [CrossRef]
22. Jamali, I.A.; Abdullah, L.; Abdul, R.M.; Dileep, K.S. Energy aware task assignment with cost optimization in mobile cloud computing. *Int. J. Commun. Netw. Syst. Sci.* **2018**, *11*, 175.
23. Liu, F.; Shu, P.; Lui, J.C. AppATP: An energy conserving adaptive mobile-cloud transmission protocol. *IEEE Trans. Comput.* **2015**, *64*, 3051–3063.
24. Lakhan, A.; Sajnani, D.K.; Tahir, M.; Aamir, M.; Lodhi, R. Delay sensitive application partitioning and task scheduling in mobile edge cloud prototyping. In *International Conference on 5G for Ubiquitous Connectivity*; Springer: Cham, Switzerland, 2018; pp. 59–80. [CrossRef]
25. Liu, F.; Shu, P.; Jin, H.; Ding, L.; Yu, J.; Niu, D.; Li, B. Gearing resource-poor mobile devices with powerful clouds: Architectures, challenges, and applications. *IEEE Wirel. Commun.* **2013**, *20*, 14–22. [CrossRef]
26. Kiran, N.; Pan, C.; Wang, S.; Yin, C. Joint resource allocation and computation offloading in mobile edge computing for SDN based wireless networks. *J. Commun. Netw.* **2019**, *22*, 1–11. [CrossRef]

27. Shah, S.D.A.; Gregory, M.A.; Li, S.; Fontes, R.D.R. SDN Enhanced Multi-Access Edge Computing (MEC) for E2E Mobility and QoS Management. *IEEE Access* **2020**, *8*, 77459–77469. [CrossRef]
28. Lakhan, A.; Li, X. Mobility and fault aware adaptive task offloading in heterogeneous mobile cloud environments. *EAI Endorsed Trans. Mob. Commun. Appl.* **2019**, *5*, e4.
29. Powell, C.; Desiniotis, C.; Dezfouli, B. The Fog Development Kit: A Platform for the Development and Management of Fog Systems. *IEEE Internet Things J.* **2020**, *7*, 3198–3213.