Article

Anomaly Detection Based Latency-Aware Energy Consumption Optimization For IoT Data-Flow Services

Yuansheng Luo [1,*], Wenjia Li [2,*] and Shi Qiu [3]

1 School of Computer and Communication Engineering, Changsha University of Science and Technology, Changsha 410114, China
2 Department of Computer Science, New York Institute of Technology, New York, NY 10023, USA
3 School of Economics and Management, Changsha University, Changsha 410022, China; 121601049@csu.edu.cn
* Correspondence: luodyx@msn.com (Y.L.); wli20@nyit.edu (W.L.)

Received: 30 October 2019; Accepted: 19 December 2019; Published: 24 December 2019

Abstract: The continuous data-flow application in the IoT integrates the functions of fog, edge, and cloud computing. Its typical paradigm is the E-Health system. Like other IoT applications, the energy consumption optimization of IoT devices in continuous data-flow applications is a challenging problem. Since the anomalous nodes in the network will cause the increase of energy consumption, it is necessary to make continuous data flows bypass these nodes as much as possible. At present, the existing research work related to the performance of continuous data-flow is often optimized from system architecture design and deployment. In this paper, a mathematical programming method is proposed for the first time to optimize the runtime performance of continuous data flow applications. A lightweight anomaly detection method is proposed to evaluate the reliability of nodes. Then the node reliability is input into the optimization algorithm to estimate the task latency. The latency-aware energy consumption optimization for continuous data-flow is modeled as a mixed integer nonlinear programming problem. A block coordinate descend-based max-flow algorithm is proposed to solve this problem. Based on the real-life datasets, the numerical simulation is carried out. The simulation results show that the proposed strategy has better performance than the benchmark strategy.

Keywords: internet of things; fog computing; E-Health monitoring system; anomaly detection; latency awareness; energy efficient; mixed integer nonlinear programming

1. Introduction

The promising big data applications based on IoT produced so much data [1,2], and thus it is impractical to transfer all these data to the data center for processing in real time. To address these challenges, the fog computing and edge computing are proposed in recent years as the distributed cloud computing solution for IoT applications. Due to the limited computation and communication capability of IoT end devices, some extensive computing models should be provided for processing large amount of IoT data. Fog computing is one of promising technologies that provide computation and communication services to IoT applications [3,4]. The concept of fog computing is similar to the edge computing [5–9]. Both of them are devoted to provide computation and communication resources for IoT users in the proximate area of IoT devices. According to the definition of European Telecommunications Standards Institute (ETSI), Multi-access Edge Computing (MEC) is one of the key technologies towards 5G and characterized by several merits such as ultra-low latency, high bandwidth, location awareness, etc. [5]. The edge computing nodes (MEC nodes) are deployed at many locations with access points, such as at the macro LTE base stations, at a multi-Radio Access Technology cell aggregation site, etc. [5]. On the other side, the fog computing nodes are not necessarily deployed around these access points. To distinguish these two concepts, in this paper, nodes with sufficient communication and computing resources scattered among IoT devices are referred to as **fog nodes**. The nano-server clusters deployed at the access points are referred to as **MEC nodes**, which usually have more resources than fog nodes. The collaborative computing of the IoT-end nodes, the fog nodes, the MEC nodes and the data center are referred to as **IoT-Fog-Edge computing** in this context.

The studies on collaboration of Fog, MEC, and Cloud computing usually focus on finding an optimal solution to allocate the IoT tasks to appropriate virtual machines which are hosted on fog, MEC or cloud nodes. In [10], the authors propose to allocate the workload among local MEC servers, neighborhood MEC servers or cloud servers to minimize the energy consumption of MEC nodes subject to delay constraints. Lyapunov drift-plus-penalty-based dynamic queue evaluation is used for the online allocation algorithm. In [11], an optimal algorithm is put forward to determine that the tasks should be allocated to clouds near the end devices or to the one far from the end devices for the energy efficient big data processing. The delay constraints to tasks in near clouds and far cloud are taken into account. In [12], an optimal algorithm for joint task allocation among mobile devices, the computing access point and the remote cloud is proposed, where computing access point can be treated as an MEC node. The studies introduced above are not correlated with the continuous data-flow (CDF) problem, which is the main concern of our work. The CDF is the data-flow continuously generated by IoT end nodes and is to travel through the fog nodes and MEC nodes before it reaches the cloud data center. The optimization to CDF problem is an optimization in multi-stage graph. On the contrary the optimization problem introduced in above studies is one-stage optimization problem. The typical CDF application in IoT-Fog-Edge computing is the E-Health Monitoring System [13–16], which will be elaborated as the Motivation Scenario in section II. Some efforts were devoted to the performance measurement and optimization of E-Health system [13–16] from the perspective of architecture and deployment optimization. To the best of our knowledge, there is little work required to optimize the performance of E-Health monitoring system from the perspective of mathematical modelling and programming.

To optimize the energy consumption of IoT devices while subject to the latency constraints, the anomaly should be discovered because the anomalous nodes would cause abnormal latencies and job loss or task failure on transmission paths [17,18]. As a result, additional retries and retransmissions will occur, resulting in increased energy consumption. Anomaly detection is a conventional method in the wireless sensor networks (WSN) [19–22], which is organized using the ad hoc way. The ad hoc mechanism makes the whole WSN system vulnerable to the intrusion of malicious nodes, thus the anomaly detection is carried out for finding the anomaly. The deficiency of anomaly detection in WSN is that many messages may be generated and exchanged in the networks. On the other hand, the network topology of fog and MEC computing is relatively stable and the nodes have more computation resources. Some systematic security and safe mechanism could be adopted in the IoT fog/edge computing, such as the block chain [4], intrusion detection system (IDS) [23], trust management scheme [24], and action-oriented programming model [9], etc. Although these security and safe mechanisms are capable of handling the most of malicious attacks, some anomalies still exist, such as the anomalies caused by hardware/software errors of fog/MEC nodes, the anomalies caused by network congestion and jitter, or the attacks which are hard to be defended such as the Denial of Service (DoS) attack [23]. To carry out the latency awareness for the CDF problem, we put forward a lightweight anomaly detection strategy. This strategy only makes use of the cumulative historical latency data of fog/MEC nodes to discovery the anomalous nodes. The results of the anomaly detection will be fed into the proposed optimization algorithm for latencies evaluation.

Many researchers modeled the energy consumption optimization problem in edge and fog computing as a mixed integer nonlinear programming problem (MINLP) [25]. This is a kind of problem for which it is difficult to find the optimal solution. Many researchers use the block coordinate descent (BCD) method to solve this kind of problem [26,27]. In these research works, the BCD method showed good performance and can quickly find the solution of complex problems. In this paper, we also use the framework of BCD method to transform the original problem into the minimum cost maximum flow sub-problem and the power control sub-problem. The proposed method is called BCDM algorithm in this paper.

In this work, we put forward a latency-aware energy-efficient **IoT-Fog-Edge Computing(IFEC)** strategy for **Continuous Data-Flow (CDF)** services. The main contributions of this work are:

- We developed a formal model for energy-efficient CDF optimization. This is a model that is composed of four level entities in IFEC computing. This model is used to formulate an optimal problem that minimizes the energy consumption subject to the latency constraints and with the anomalous fog or MEC nodes existing in systems.
- We proposed a novel lightweight anomalous nodes detection strategy for latency-aware CDF optimization.
- We designed a block coordinate descend-based max-flow algorithm to solve latency-aware energy-efficient CDF problem iteratively.
- The performance of proposed model and algorithm was evaluated by simulations based on real-life datasets.

The remainder of this paper is organized as follows: In Section 2, the motivation scenarios in E-Health system are elaborated. In Section 3 we present the system model and problem formulation. In Section 4, we put forward the proposed solutions for the CDF problem. The numerical simulation based on real-life datasets are presented in Section 5 and we draw conclusions in Section 6. The acronyms used in this paper are listed in Table 1.

Table 1. Notations.

Symbol	Description
U	Utility function
I	the amount of IoT-End nodes
M	the amount of MEC nodes
J	the amount of Fog nodes
λ	arrival rate of tasks
τ	latency constraint
μ	service rate of fog and MEC nodes
B	bandwidth of wireless channel
H	channel gains
σ^2	Gaussian white noise

2. Motivation Scenario

Thanks to the rapid progress of wireless communications and wearable devices, the E-Health Monitoring (EHM) System has become a paradigm of IoT applications [13–16]. A typical EHM system is composed of an IoT sensing subsystem, networking subsystem, cloud data processing and storage subsystem. In [13], the EHM system is divided into four parts, which are wearable devices, Machine-to-Machine (M2M) gateway, Network Service Capability Layer (NSCL), Data processor and openEHR services. Although the authors did not present the IoT-Fog-Edge computing, it actually can be treated as an CDF scenario because the proposed model has the same characteristics as CDF. The M2M gateways continuously collect data from sensors (such as the heart rate, blood pressure, blood oxygen saturation, etc.) and send the data periodically to data processors in the virtual machines hosted at a cloud provider. The data flow will go across the M2M gateways, NSCL and reach the cloud at last. In [14–16], the IoT, fog, edge and cloud integrated architecture for EHM system was clearly illustrated. The performance, including the latency, availability, and the potential challenges in EHM systems, were addressed in [13–15] respectively. A common characteristic in the EHM system is that the data flow starting from the IoT devices will travel across the fog layer and edge layer before it reaches the cloud data center. The network latency and availability are mainly influenced by the devices in networking subsystems such as the fog and MEC nodes. We generalized the system model of EHM system to the continuous data-flow IoT system. In our model, each IoT device has its specific domain tasks, such as the EHM tasks. The data flow generated by different IoT devices can go through same or different network paths. Our aim is to minimize the global energy consumption of multiple IoT devices while subject to the latency constraints at each IFEC level.

3. System Model

In CDF application scenarios, nodes in the networks could be categorized into following four levels according to their computation capability:

1. **IoT end level**: The IoT end devices belongs to this level, such as sensors, RFID, etc. which are used to collect the raw data in IoT.
2. **Fog level**: The fog nodes with very limited communication and computation capability belong to this level, such as the IoT gateway.
3. **MEC level**: The server clusters with limited computation capability belong to this level, which are deployed at places in proximity to access points of mobile networks, such as at the macro LTE base stations, at a multi-Radio Access Technology cell aggregation site, etc. [5].
4. **Cloud level**: There are only data centers at this level.

As previously mentioned, in the continuous data flow service scenarios, the IoT end devices should send the data to Fog nodes or MEC servers for preprocessing before these data are sent to data centers. A client software should be installed on end devices to communicate with the Fog nodes and MEC servers to receive the application services. These applications have domain-specific tasks that are offloaded to Fog nodes or MEC servers to execute all kinds of complicated computation, e.g., intelligent video acceleration, augmented reality (AR), etc. The computation resources are actually virtual machines that are deployed specifically for users' applications. The VMs are referred to as *Proxy VMs* [28].

Without loss of generality, we assume the data traffic starting at the end devices should go through the fog nodes to the MEC servers and reach the data center at last. In the case that the end device does not need to offload data to the MEC server, but directly sends data to the data center by the fog node, the MEC node can be merely treated as an access point to the core networks. Our aim is to minimize the energy consumption of end devices while subject to the latency constraint in each level.

The system framework can be formulated as a tuple (UE, FN, EC, DC). UE is a collection with I IoT end devices. $UE = \{ue_i, i = 1, 2, \ldots, I\}$. FN is a collection with J fog nodes, $FN = \{fn_j, j = 1, 2, \ldots, J\}$. EC is a collection with M MEC nodes, $EC = \{ec_m, m = 1, 2, \ldots, M\}$. DC is the data center. Each ue_i can be expressed as a triple tube: $ue_i = \{\lambda_i, P_i, \tau_i\}$, where λ_i is the arrival rate of tasks of ue_i in a time unit. We assume tasks arrive at ue_i according to the poisson distribution. The value of arrival rates of Poisson distribution can be estimated by fitting method [29]. P_i is the wireless transmission power of ue_i and τ_i is the maximum acceptable latency for finishing a task to ensure the quality of services(QoS). fn_j and ec_m can be expressed as tubes $fn_j = \{\mu_j, pr_j\}$ and $ec_m = \{\mu_m, pr_m\}$ respectively, where μ is the service rate or processing capability, i.e. the expected number of tasks which can be processed in a time unit. We assume service time of a fog node or an MEC node follows exponential distribution with mean $\frac{1}{\mu}$. pr is the reliability that a task can be executed successfully.

3.1. Anomalous Nodes Discovery and Confidence Evaluation

Latency variation in IoT may be caused by network congestion or jitter. The slight variation would not change the regularity of the statistic features of network latency. On the contrary, the software or hardware errors, or attacks from the malicious users, may cause the anomalies of fog nodes and MEC nodes. Thus, the network performance of these nodes may become unstable and unpredictable. To guarantee the QoS of data flow services, an optimal solution for CDF problem should bypass these anomalous nodes. We put forward a lightweight anomalous nodes discovery strategy. This strategy is based on the observation that the anomalous nodes will exhibit some anomalous behaviors and deviate from the statistic regularity of normal nodes.

We give following definition of anomalies according to the definition of Das et al. [30].

Definition 1. *In our paper, anomalies are defined as any observations of latencies that are different from the normal behavior of the latency data.*

3.1.1. Chi-Square Test and Similarity Measurement

χ^2 test is a test of the goodness-of-fit [31]. It is used to test the null hypothesis that the observed data comes from a specific distribution. Given the sample size is large enough, we have following null Hypothesis 1:

Hypothesis 1 (H1). *The latency value of the observed fog or MEC node comes from the normal distribution.*

This hypothesis derives from the research on latency characteristics of mobile IoT [13]. If a fog node or MEC node is disrupted, its statistic feature of latency should be different with the normal distribution. Let $p_i(t)$ be the p-value got from tth round χ^2 test against latency sample $\ell_i(t)$ of ith node. $\ell_i(t) = (s_i(0), s_i(1), \ldots, s_i(t))$ is a sample vector and $s_i(t)$ is the tth latency sample of node i. $\varphi_i(t) = (p_i(0), p_i(1), \ldots, p_i(t))$ is the vector of p-value. The Cosine similarity of latency distribution of two nodes is defined as follows:

$$sim(i,j) = \frac{\varphi_i \cdot \varphi_j}{\|\varphi_i\| \times \|\varphi_j\|} \tag{1}$$

Let the node j be a normal node which is selected as a *guard* by the service provider of data flow services. If the node i is also a normal node, it should show similar behaviors with i from the perspective of latencies. The similarity can be measured by the cosine function $sim(i,j)$. Those nodes of which behaviors deviate from the baseline behaviors can be treated as anomalous nodes. The concept *guard node* is used in the anomaly detection and monitoring in wireless sensor networks (WSN) [18,32]. Guard node is a kind of node used for anomaly detection in wireless sensor networks (WSN). The guard node is between the sending node and the receiving node, which can be used for normal communication and monitoring. The selection of guard nodes is based on the location and trustworthiness of the nodes [18,33,34]. Because the continuous data flow network in this paper is not a WSN-like mobile ad-hoc network, so when selecting guard nodes, we do not need to consider the location factor, but only need to select according to the trustworthiness of the nodes. Including the architecture of nodes, security level and other trust related attributes of nodes can be used as a measurement of trustworthiness. The guard nodes in WSN should be responsible for monitoring and detecting the anomalies in addition to serving as the baseline. Nevertheless, the guard nodes in our work merely are treated as the baseline. The issues on monitoring and detecting malicious nodes are out of scopes of this work.

3.1.2. F-Test

In some cases, the behaviors of a node may conform to normal distribution but the node does not have the same parameters of distributions as the guard node. An F-test is used to test whether two samples come from the normal distribution with the same variance. In this context, the variance of the latency vector of a node is compared to the one of the guard node by F-test. If the p-value of the test is less than 0.05, we reject the null Hypothesis 2.

Hypothesis 2 (H2). *The latencies of the observed fog or MEC node and of the guard node come from the normal distributions with the same variance.*

3.1.3. Put It All Together

According to the results from Sections 3.1.1 and 3.1.2, we can calculate the reliability value. The real value corresponding to different reliability level depends on the application in practice. For example, 0.9 can be treated as a high value of reliability for most business applications. However, applications in finance and banking industry require more than 0.99 reliability. A simple way to estimate the real value is to calculate it by the value of the guard node j as follows:

$$\begin{aligned} pr_i = {} & IND(\lfloor \frac{sim(i,j)}{0.5} \rfloor)(\frac{1}{F_t(i,j)} pr_j) \\ & + (1 - IND(\lfloor \frac{sim(i,j)}{0.5} \rfloor)) sim(i,j) pr_j \end{aligned} \tag{2}$$

where $\lfloor x \rfloor$ is to get the largest integer value less than/equal to x. $IND(x) = 1$ when $x \geq 1$ and $IND(x) = x$ otherwise. $F_t(i,j)$ is the F-test for node i and j. When the number of latency samples n approaches $+\infty$, the degree of freedom also approaches $+\infty$, and $F_t(i,j)$ should approach 1 given the node i is as confident as guard node j. It should be noted that our aim is not to get the accurate value of reliability of a node. Our aim is to find the anomalous nodes and bypass these nodes in the optimal solution of the CDF problem given there are alternative normal nodes.

Figure 1 is used to illustrate the execution process of proposed anomaly detection algorithm and its position in the whole optimization model.

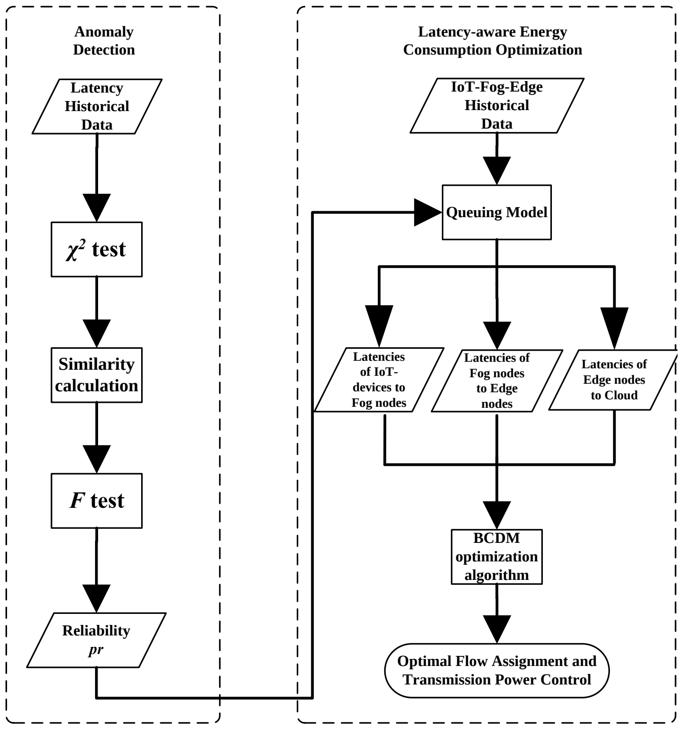
Anomaly Detection
Latency Historical Data
χ² test
Similarity calculation
F test
Reliability pr
Latency-aware Energy Consumption Optimization
IoT-Fog-Edge Historical Data
Queuing Model
Latencies of IoT-devices to Fog nodes
Latencies of Fog nodes to Edge nodes
Latencies of Edge nodes to Cloud
BCDM optimization algorithm
Optimal Flow Assignment and Transmission Power Control

Figure 1. Flowchart of the optimization procedure of CDF.

3.2. Tandem Queue Model

We put forward the tandem queue model depicted in Figure 2 for investigating the execution latency of tasks.

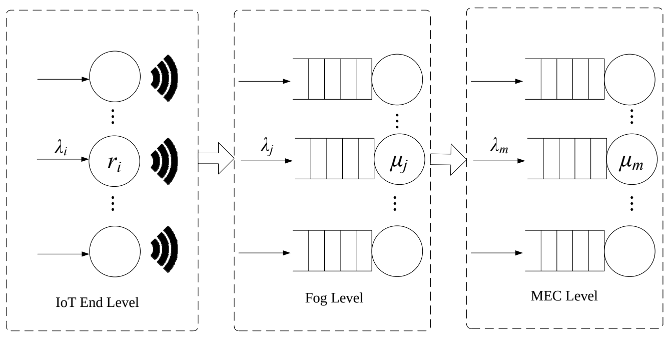

Figure 2. The tandem queue model of IoT-Fog-Edge Computing.

3.2.1. Latency in IoT End Level

In the IoT end level, tasks arrive at end equipment ue_i according to the poisson distribution with arrival rate λ_i. Without loss of generality, we assume the data size of each task is equal to d. The transmission latency at this level can be expressed as following equations.

$$T_i^{comm} = \frac{1}{\frac{r_i}{d} - \lambda_i}, i = 1, 2, \ldots, I \tag{3}$$

$$r_i = B\log(1 + \frac{P_i H_i}{\sigma^2}) \tag{4}$$

where B is the wireless channel bandwidth. P_i is the transmission power. σ^2 is the variance of complex white Gaussian channel noise. H_i is the average channel gain. r_i is the average transmission data rate. $\frac{r_i}{d}$ is the number of tasks which can be transmitted in a time unit through the wireless channel. The transmission queue of each end equipment is an M/M/1 queue. To guarantee the queue stability, following constraint must be satisfied:

$$\lambda_i < \frac{r_i}{d}, i = 1, 2, \ldots, I \tag{5}$$

3.2.2. Latency in Fog and MEC Level

Same as the model at the IoT end level, the latency in fog level and MEC level can also be expressed as the queue model as follows:

$$T^{IoT \to fog} = \frac{x_{ij} d}{r_i} \leq \tau_1 \tag{6}$$

$$T_j^{fog} = \frac{1}{pr_j}\left(\frac{1}{(\mu_j - \lambda_j)}\right) \leq \tau_2 \tag{7}$$

$$\mu_j > \lambda_j \tag{8}$$

$$\sum_{i=1}^{I} x_{ij} = \lambda_j, j = 1, 2, \ldots, J \tag{9}$$

where x_{ij} is the number of tasks sent from the ue_i to the fn_j. $\frac{x_{ij} d}{r_i}$ is the transmission duration that the ue_i transmits $x_{ij} d$ units data to fn_j.

$$T_m^{mec} = \frac{1}{pr_m}\left(\frac{1}{\mu_m - \lambda_m}\right) \leq \tau_3 \tag{10}$$

$$\mu_m > \lambda_m \tag{11}$$

$$\sum_{j=1}^{J} x_{jm} = \lambda_m, m = 1, 2, \ldots, M \tag{12}$$

The pr_j and pr_m are the evaluated reliability that a task can be executed successfully by a node. As previously mentioned, we measure the confidence of the nodes in our model and map the confidence to the reliability. According to the binomial distribution, the expectation of the number of retrying for one successful execution is $\frac{1}{pr}$.

3.3. Problem Formulation

The latency-aware energy-efficient CDF problem can be formulated as following expression:

$$\textbf{MP}: \underset{x_{ij}, P_i}{Minimuze}\ U = \sum_{i=1}^{I} \sum_{j=1}^{J} E_i x_{ij} \tag{13}$$

$$\text{Subject to :}$$

$$(3) \sim (12)$$

$$x_{ij} \in \{0, 1, 2, \cdots, x^{max}\} \tag{14}$$

$$P_i \in [P^{min}, P^{max}] \tag{15}$$

where $E_i = \frac{P_i}{r_i} d$. We assume that an IoT node can only be connected to one access point in a duration. Thus, transmission power should be same for the same IoT node. So is the transmission rate. The problem **MP** is a mixed-integer-non-linear programming problem (MINLP), which is an NP-hard problem [35]. There is one multi-dimension combinatorial decision variable x and one continuous multi-dimension decision variable P. There is no generic optimal algorithm to solve this kind of problem in polynomial time complexity. In following section, we put forward two approximate algorithms, the block coordinate descent based multi-flow algorithm (BCDM) and Best-effort algorithm.

4. Solutions

4.1. Block Coordinate Descent Based Multi-Flow Algorithm

In the conventional block coordinate descent method, the vector of variables is partitioned into different blocks. The sub-function over each block with all other blocks fixed is assumed to be differentiable and convex [36,37]. Nevertheless, the original problem **MP** does not satisfy these assumptions. Thus, we have to relax the original problem **MP**. First, we fix the integer variables x to get the following sub-problem:

$$\textbf{SP1}: \underset{P_i}{Minimuze}\ U(P_i) = \sum_{i=1}^{I} \sum_{j=1}^{J} \frac{P_i d}{r_i} x_{ij} \tag{16}$$

$$\text{Subject to : } (3) \sim (6)$$

$$P_i \in [P^{min}, P^{max}] \tag{17}$$

where constraints (4)–(6) can be further reduced to following expression:

$$P_i \geq \frac{(2^{\frac{x_{ij}d}{B\tau_1}} - 1)\sigma^2}{H_i}, j = 1, \cdots, J \tag{18}$$

The objective function of **SP1** is not meant to be convex. However, according to the research in [37], if a function F is differentiable and strictly quasi-convex over each block, its limit point is a critical point. Thus, we only need to confirm that $U(P_i) = \sum_{i=1}^{I}\sum_{j=1}^{J} \frac{P_i d}{r_i} x_{ij}$ is quasi-convex. We have following proposition:

Proposition 1. *Let $F(P_i) = \frac{P_i}{r_i}$, F is quasi-convex. Where P_i and r_i are the transmission power and data rate respectively.*

Proof. At first, we calculate the first order derivative of F.

$$\begin{aligned} \frac{dF}{dP_i} &= \frac{\ln 2}{\ln(aP_i + 1)} - \frac{aP_i \ln 2}{(aP_i+1)\ln^2(aP_i+1)} \\ &= \frac{\ln 2}{\ln(aP_i+1)}(1 - g(P_i)) \end{aligned} \tag{19}$$

$$g(P_i) = \frac{aP_i}{(aP_i+1)\ln(aP_i+1)} \tag{20}$$

where $a = \frac{H_i}{\sigma^2}$. aP_i is actually the signal-to-noise ratio (SNR). Let $aP_i + 1 = x$, $g(P_i) \Rightarrow g'(x) = \frac{x-1}{x \ln x}$. We have the following result:

$$\forall a > 0, \lim_{P_i \to 0} g(P_i) = \lim_{x \to 1} g'(x) = \lim_{x \to 1} \frac{x-1}{x \ln x} \tag{21}$$

$$= \lim_{x \to 1} \frac{1}{\ln x + 1} = 1 \tag{22}$$

Then we calculate the first order derivative of $g(P_i)$:

$$\frac{dg(P_i)}{dP_i} = \frac{a(\ln(aP_i+1) - aP_i)}{(aP_i+1)^2 \ln^2(aP_i+1)} \tag{23}$$

$\frac{dg(P_i)}{dP_i} < 0$ when $P_i > 0$, thus $g(P_i)$ is **monotonic decreasing**. Hence according to (21) and (22), $g(P_i) < 1$ will always hold with the $P_i > 0$. Thus, $\frac{dF}{dP_i} > 0$ always holds too, which means $F(P_i)$ **is monotonic increasing**. Suppose P_1 and P_2 are two transmission power values, $P_1 \geq P_2$. $\forall \alpha \in [0, 1]$, we can get following result:

$$\begin{aligned} F(\alpha P_1 + (1-\alpha)P_2) &\leq F(\alpha P_1 + (1-\alpha)P_1) \\ &= F(P_1) = max(F(P_1), F(P_2)) \end{aligned} \tag{24}$$

This result means $F(P_i)$ is quasi-convex. □

According to the Proposition 1, the $U(P_i)$ is also quasi-convex because it is a polynomial of $F(P_i)$.

Then we consider the sub-problem in which continuous variables are fixed. The sub-problem can be expressed as follows:

$$\textbf{SP2}: \underset{x_{ij}}{Minimuze}\ U = \sum_{i=1}^{I}\sum_{j=1}^{J} \frac{P_i d}{r_i} x_{ij} \tag{25}$$

Subject to : (7) $\sim$ (12)

$$x_{ij} \in \{0, 1, 2, \cdots, x^{max}\} \tag{26}$$

The **SP2** is an integer programming problem, which is not convex or quasi-convex. Hence we have to transform it to a convex problem.

Constraints (7)∼(12) can be transformed as follows:

$$0 \le x_{ij} \le \tau_1 \frac{r_{ij}}{d} \tag{27}$$

$$0 \le x_{jm} \le \mu_j - \frac{1}{\tau_2 pr_j} \tag{28}$$

$$0 \le x_{mt} \le \mu_m - \frac{1}{\tau_3 pr_m} \tag{29}$$

$$\sum_{j=1}^{J} x_{ij} = \lambda_i \tag{30}$$

$$\sum_{i=1}^{I} x_{ij} = \sum_{m=1}^{M} x_{jm} \tag{31}$$

$$\sum_{j=1}^{J} x_{jm} = x_{mt} \tag{32}$$

$$i = 1, \cdots, I; j = 1, \cdots, J; m = 1, \cdots, M; t = 1$$

where the index variable t represents the data center. This is a kind of linear relaxation. After the linear relaxation, the **SP2** can be solved by linear programming solver in polynomial time-complexity steps. Nevertheless, the solution cannot be ensured to be integral. To guarantee the solution is integral, following transformation should be carried out.

As depicted in Figure 3, a dummy node **S** is added to the IFEC model as a source node, which has some dummy edges to be connected with the IoT end nodes. Each fog node or MEC node are split into two nodes connected by a dummy edge with a weight, which is the upper bound of processing capability of the corresponding node. A dummy edge connecting the **S** with **T** is added. If some tasks could not be offloaded to fog nodes, they will be sent to **T** directly by this dummy edge. This dummy edge has almost infinite throughput(such as the 5G communication channel) but very high energy consumption. If a task went through this dummy edge, it means the failure of receiving fog computing service and it is not a part of solution. Thus, the CDF model can be converted to a Minimum Cost Maximum Flow model (MCMF). To guarantee that the solution is integral, the parameters should be integers, which can be achieved by rounding and scaling. For example, if the capacity of a node is 4.2, it can be rounded to 4. If the energy consumption is 0.3 Joule, it can be scaled up to 300 Millijoule.

After the above linear relaxation and transformation are conducted, the original problem **MP** was converted to two convex (or quasi-convex) sub-problems, thus it can be solved by block coordinate descent algorithm iteratively. The pseudo codes of Block Coordinate Descent-based Multi-flow algorithm (BCDM) is illustrated in Algorithm 1.

Based on the above analysis, we can get the following conclusions about the effectiveness of BCDM algorithm:

Since every subproblem of CDF optimization is differentiable, convex and strictly quasi-convex, the BCDM algorithm can converge to the local optimal solution.

After the linear relaxation of the subproblem SP2, we use the simplex method to solve the linear programming problem. Because the worst-case time complexity of the simplex method is exponential, so the worst-case time complexity of the BCDM algorithm is exponential. However, in general, simplex algorithm is an efficient algorithm, which can get the solution in polynomial time. Therefore, in general, BCDM algorithm is polynomial time complexity algorithm.

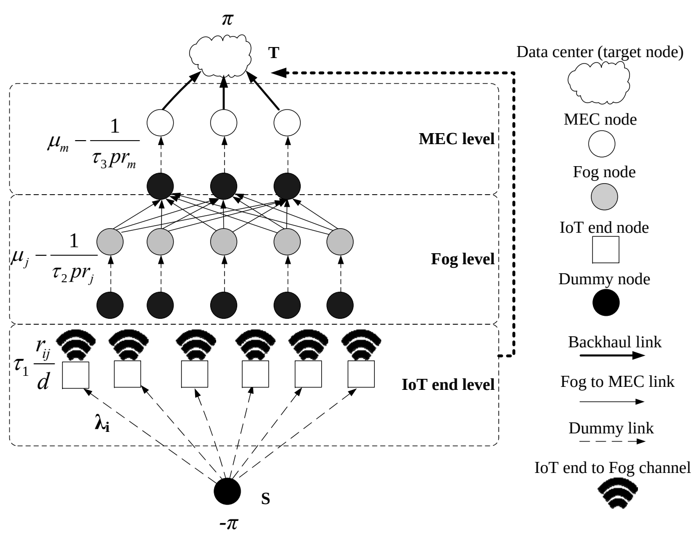

Figure 3. The Minimum Cost Maximum Flow Model for the CDF problem in IFEC.

Algorithm 1 Block Coordinate Descent based Multi-flow Algorithm

```
Input: $\lambda_i, \mu_j, \mu_m, \tau_1, \tau_2, \tau_3, H_i, B, \sigma, pr_j, pr_m$;
Output: $x, P$;
1: Initialize $P_i$ as $P^{min}$
2: while $U^{old} - U^{new} > \varepsilon$ do
3:     solve MCMF problem with $P_i$ fixed to get $x_{ij}$
4:     update each $P_i$ iteratively with $x_{ij}$ and the other $P_j$ fixed, $i \neq j$
5:     renew $U$, $P$ if $U^{new}$ descends
6: end while
7: return $x, P$
```

4.2. Best-Effort Algorithm

Since there is no same work currently to solve the CDF problem as proposed in this paper, we use the Best-effort method as the benchmark, which was used for IP data-flow service in multi-access wireless networks and can be adapted to the CDF problem easily. The Best-effort algorithm is carried out on the different level. At each level, the algorithm always tries its best to allocate the tasks to one node in the next level till it reached its upper capacity, and then turn to another node in the next level. The Best-effort algorithm is illustrated in Algorithm 2.

Algorithm 2 Best-effort Algorithm

```
Input: $\lambda_i, \mu_j, \mu_m, \tau_1, \tau_2, \tau_3, H_i, B, \sigma, pr_j, pr_m$;
Output: $x, P$;
1: for each $i$ in $I$ do
2:     for each $j$ in $J$ do
3:         allocate as many tasks of $ue_i$ as possible to $fn_j$ till reach its capacity
4:     end for
5:     calculate $P_i$ for each $ue_i$
6: end for
7: while the solution is infeasible do
8:     carry out the Best-effort strategy in fog level
9:     carry out the Best-effort strategy in MEC level
10:    if the solution is infeasible, decrease the $x_{ij}$ with largest $P_i$ unless $x_{ij}$ becomes 0
11: end while
12: return $x, P$
```

5. Simulations

In this section, two types of simulations are presented. One was conducted for evaluating the feasibility of proposed latency awareness strategy, the other was conducted for measuring the performance of proposed BCDM algorithm.

5.1. Anomaly Detection Based Latency Awareness

According to the study by Pereira et al. [13], the E-Health Monitoring (EHM) ecosystem can be divided into Gate Way (GW), Network Service Capability Layers (NSCL), Data Processor (DP) and openEHR services. They reported their real-life experiment results on evaluating the latency performance of service composition of a mobile E-Health application. *They argued that end-to-end (E2E) latency is composed of latencies between neighbor system components and the latencies that compose the E2E latency follow a normal distribution*. Part of the measured latency values are listed in Table 2.

Table 2. Latencies In EHM System.

Parameter Name	Parameter Value (Unit)
Latency between GW and NSCL	$0.9671 sec \pm 0.0186$
Latency between NSCL and DP	$0.0138 sec \pm 1.802 \times 10^{-4}$
Latency between DP and EHR	$0.3130 sec \pm 0.0470$

To test the proposed strategy of anomaly detection based reliability evaluation, we generated a sample sequence of latencies according to the normal distribution and the latency between GW and NSCL in Table 2. This sample sequence was chosen as the baseline of the anomaly detection. The other two sample sequences also were generated as the historical data of anomalous nodes. One of the sequences was the latencies of the malicious node, and generated using the same way as generating the baseline except that some anomalous data were injected into the sequence. We assumed that the malicious node would misbehave with 50% probability [38], thus at each sampling point, the malicious node may generate the normal latency values with 50% probability. In other 50% probability cases, the sample generation is failure. The sample generation will be retried with adding a waiting duration to the latency. The maximum number of retries is 4, and the waiting duration is 1, 2, 4, 8 respectively. The final latency will be set as 1000 s if the maximum retries reached. Another sample sequence is the latencies of the abnormal node. The abnormal node here refers to the node that has not been maliciously attacked although it shows abnormal behavior. This sequence was generated according to the normal distribution but with different distribution parameters. Total 1000 samples were generated for each sample sequence. The order of the samples in the sequence was treated as the rounds of consecutive sampling in the real scenario.

The simulation results are shown in Figure 4. The reliability was calculated according to Equation (1). The reliability of the baseline node is always 1. As shown in the results, the reliability of the malicious node and abnormal node fluctuated obviously in the first 50 rounds. This is because the proposed scheme is based on the central limit theorem of probability statistics. The central limit theorem works only when the number of samples is large enough. After 50 rounds, the reliability of malicious and abnormal nodes decreased gradually. Finally, the reliability of abnormal nodes was gradually stable in the interval between 0.1 and 0.2, while that of malicious nodes approach 0. Experimental results show that our scheme can effectively detect anomalous nodes. It should be noted that our purpose is not to get the real probability of normal behavior of anomalous nodes, but to get a reliable estimate value, and use this estimate value to select normal nodes first, and bypass anomalous nodes.

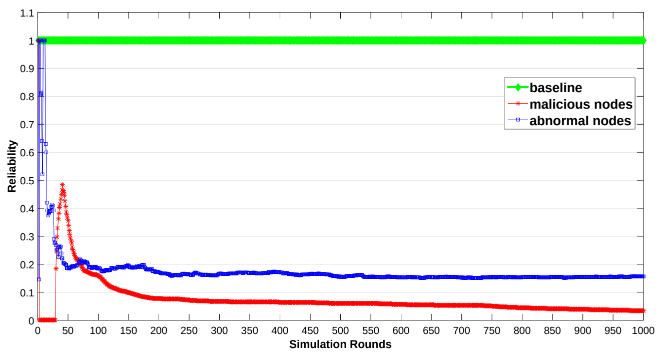

Figure 4. The Reliability Evaluation in 1000 Rounds Consecutive Sampling.

5.2. Energy Efficient Optimization for IoT Continuous Data-Flow Services

The effective of the proposed energy efficient optimization algorithm was verified in this section.

5.2.1. Task Arrival Rate Analysis

He et al. [29] studied the traffic arrival mode of the GSM base station and verified that it conforms to the Poisson distribution. The task arrival rate used in this section was set according to the real dataset provided by Barlacchi et al. [39]. There are several different datasets in their research, such as telecommunications, weather, news, social networks, etc. In our simulation, we focused on the Internet traffic activity data. Their dataset contains data collected from Milan city and province of Trentino. The areas of Milan and Trentino are divided into grids, which are composed of squares with size of about 235 $\times$ 235 m. Each square is assigned a square ID. The data set contains the amount of traffic reached in unit time (10 min) in each square. We used Poisson parameter estimates to fit the arrival rate per hour, and got the confidence interval of the hourly arrival rate with 95% confidence in one day.

The arrival rates in 24 h at 4 different squares in Trentino and Milan are illustrated in Figures 5–8. The curves drawn in these figures are the average of the arrival rate, and the error bars represent the upper and lower bounds of the confidence interval. As shown in the figures, the arrival rate will vary greatly in different hours and in different regions. The minimum value of arrival rate is less than 2, while the maximum value is close to 55. It should be noted that in [39], every time a user started an Internet connection, a record will be generated. However, it is well known that the amount of data traffic generated by each connection is different. In addition, during the connection, the user program may generate multiple tasks, and the size of each task is different too. For example, the data size of sending an instruction or a heartbeat signal does not exceed 1 KB, while the data size of sending a face recognition picture may exceed 5 MB. Therefore, for the convenience of analysis, we consider that each CDF task has the same size and does not exceed the maximum transmission unit (MTU) of TCP/IP protocol, i.e., 1500 bytes. For the records in [39], suppose 1 MB data was generated for each user connection. If the arrival rate was 10 within one hour, the data size generated was 10 MB. That was equivalent to 7000 tasks arriving in an hour in our paper.

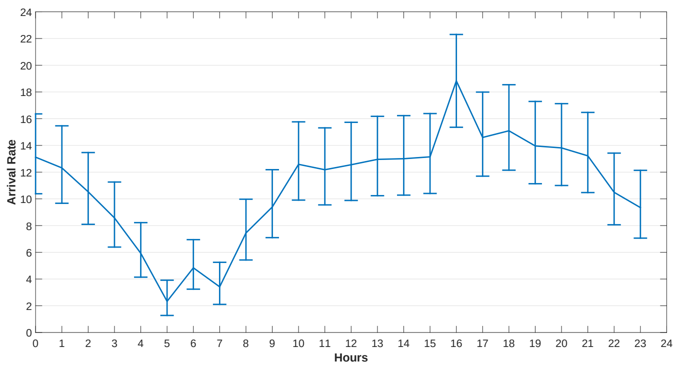

Figure 5. The Arrival Rate in 24 h, Square id 1000, Trentino, 1 November 2013.

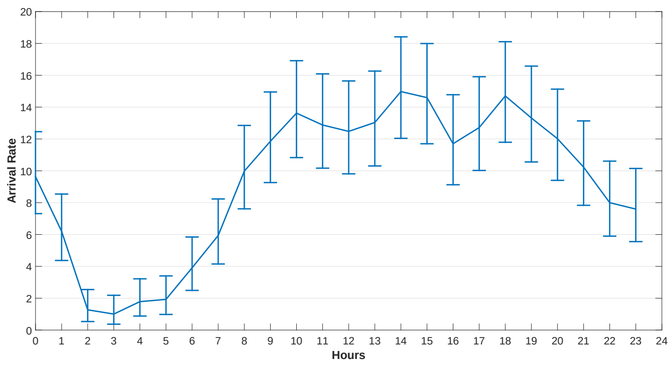

Figure 6. The Arrival Rate in 24 h, Square id 1, Milan, 1 November 2013.

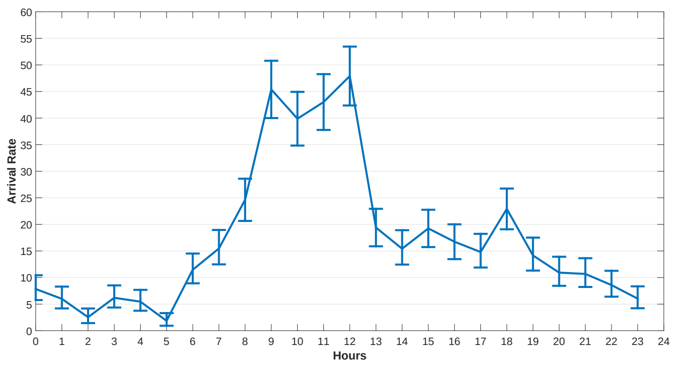

Figure 7. The Arrival Rate in 24 h, Square id 100, Milan, 1 November 2013.

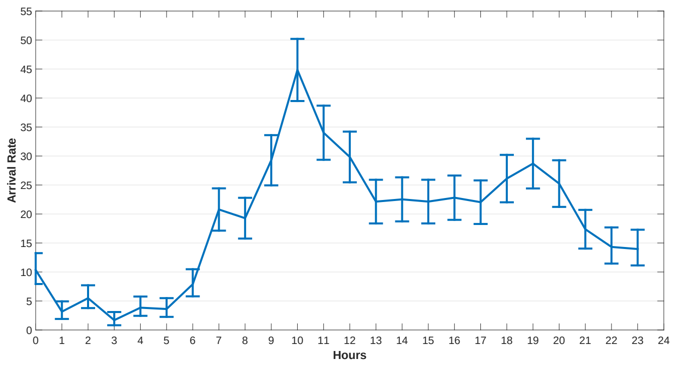

Figure 8. The Arrival Rate in 24 h, Square id 10000, Milan, 1 November 2013.

5.2.2. Verification of the Proposed Algorithm

Some of the simulation parameters in this section are listed in Table 3. The performance metrics are total energy consumption and average energy consumption (AE) of IoT end nodes. AE is calculated as following equation:

$$AE = \frac{U}{\sum_{i=1}^{I} \sum_{j=1}^{J} x_{ij}} \tag{33}$$

Table 3. Simulation Parameters.

Parameter Name	Parameter Value (Unit)
The Distance from IoT-End Nodes to Access Points	100–500 m
The Default Number of IoT-End Nodes	60
The Default Number of Fog Nodes	20
The Default Number of MEC Nodes	10
Bandwidth of Wireless Channel	1.08×10^6 Hz
Default Processing Capacity of MEC Nodes μ_m	50/sec
Default Processing Capacity of Fog Nodes μ_j	20/sec
The Data Size of a Task d	3.0×10^5 Bit
Latency Constraint τ	0.3 s
Default Task Arrival Rate of Each IoT-Fog Node λ	10/sec
The Reliability of Normal Fog Nodes	$[0.9, 0.99]$
The Reliability of Normal MEC Nodes	$[0.9, 0.99]$
Default Percentage of Malicious Fog Nodes	20%
Default Percentage of Malicious MEC Nodes	10%
Default Reliability of Malicious Fog Nodes	0.2
Default Reliability of Malicious MEC Nodes	0.2
Background Noise σ^2	-100 (dBm)

We studied the impact of the number of MEC nodes, the impact of the number of fog nodes and the impact of percentage of malicious fog nodes in this section. The results are shown in Figures 9–17 respectively. As shown in the results, both BCDM algorithm and Best-effort algorithm showed good scalability when the simulation parameters changed, especially in the presence of malicious nodes. This is because the latency-awareness strategy based on anomaly detection proposed in this paper can effectively bypass the anomalous nodes, thus reducing energy consumption. In all cases, BCDM algorithm showed better performance than Best-effort algorithm. It should be noted that a line chart and a bar chart are used to show the number of tasks allocated in Figures 11 and 14. This is because the number of tasks assigned by BCDM and Best-effort algorithm is the same, and different graphical methods are used to show them more clearly.

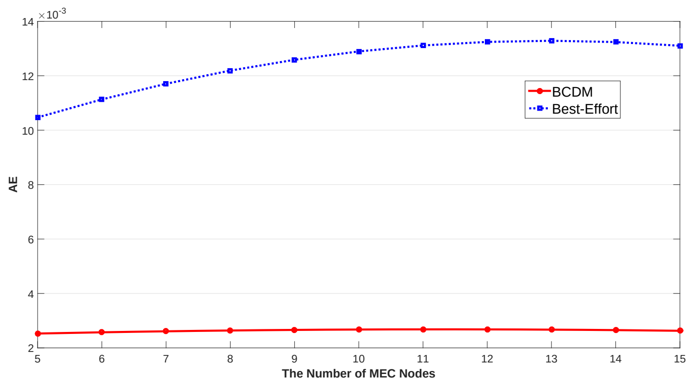

Figure 9. The Impact of the Number of MEC Nodes to Average Energy Consumption.

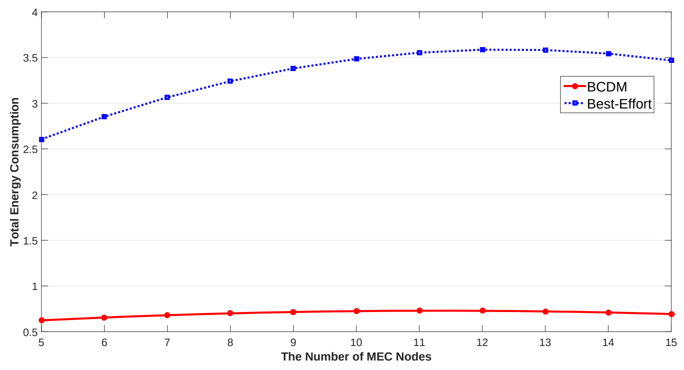

Figure 10. The Impact of the Number of MEC Nodes to Total Energy Consumption.

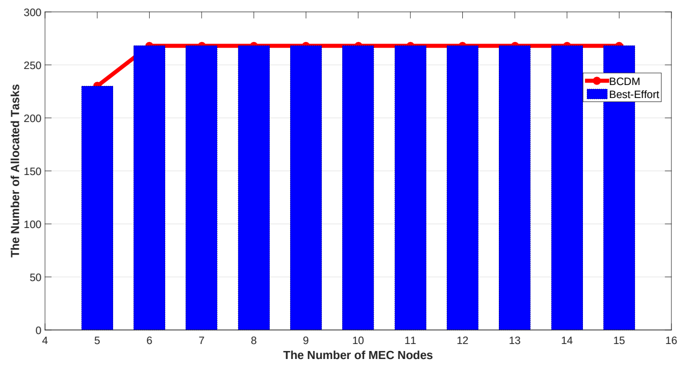

Figure 11. The Impact of the Number of MEC Nodes to Total Number of Allocated Tasks.

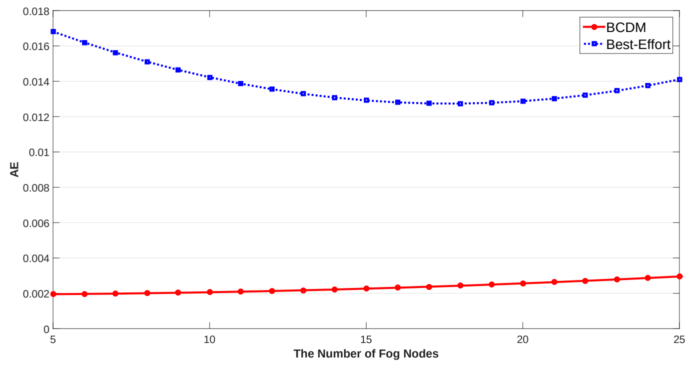

Figure 12. The Impact of the Number of Fog Nodes to Average Energy Consumption.

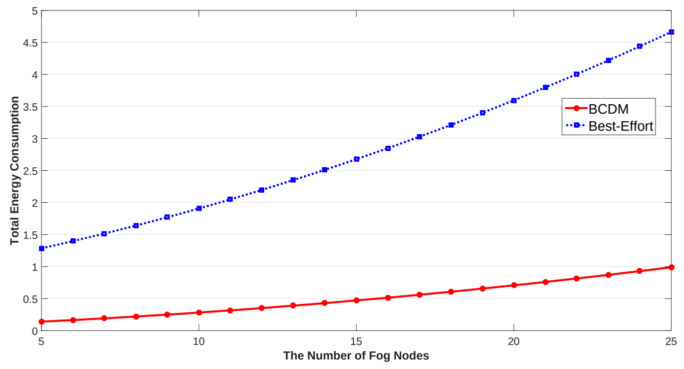

Figure 13. The Impact of the Number of Fog Nodes to Total Energy Consumption.

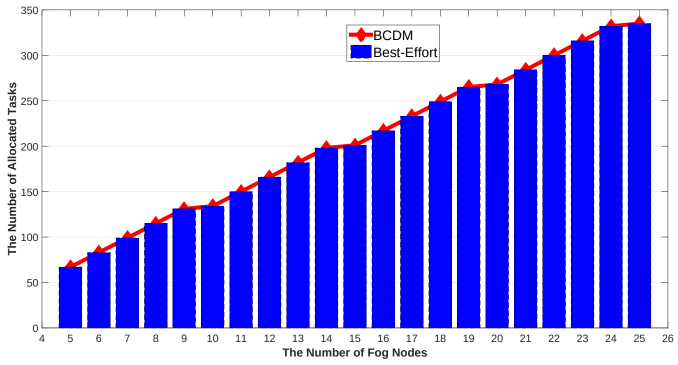

Figure 14. The Impact of the Number of Fog Nodes to Total Number of Allocated Tasks.

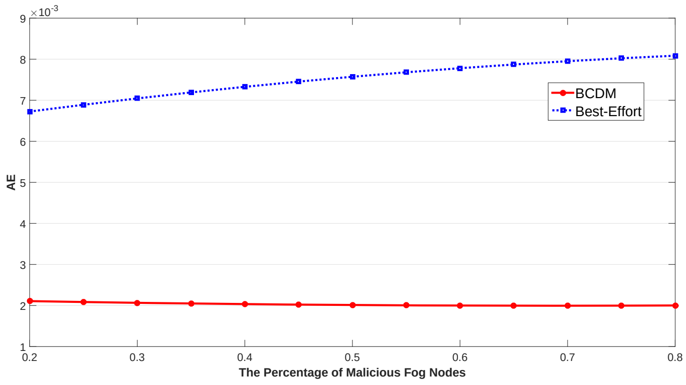

Figure 15. The Impact of Percentage of Malicious Fog Nodes to Average Energy Consumption.

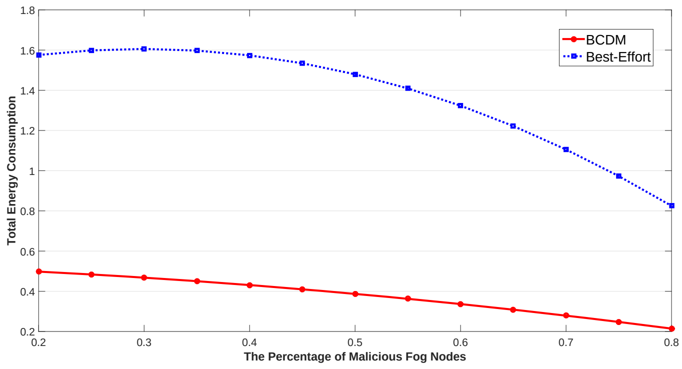

Figure 16. The Impact of Percentage of Malicious Fog Nodes to Total Energy Consumption.

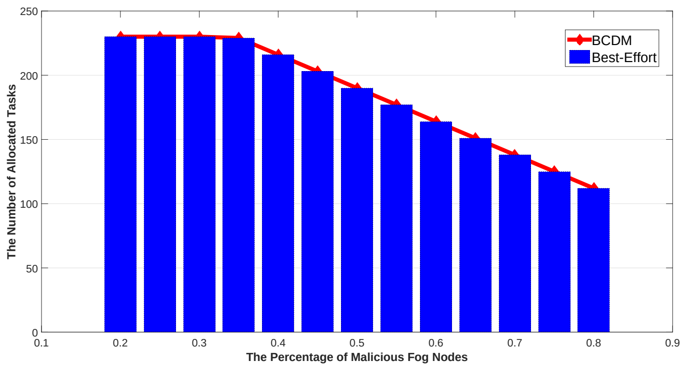

Figure 17. The Impact of Percentage of Malicious Fog Nodes to Total Number of Allocated Tasks.

The impact of the number of fog nodes and MEC nodes on energy consumption is shown in Figures 9–14. As depicted in these figures, the impact of the number of fog nodes on energy consumption was greater than that of MEC nodes. This is because in our simulation parameter setting, MEC node has stronger task processing ability, which is also one of the main differences between fog nodes and MEC nodes. Therefore, in the simulation, the main bottleneck of performance was exhibited in the fog node layer. From the simulation results, it can be found that the change of MEC node number had little effect on energy consumption, while the increase of fog node number made the decrease of energy consumption very obvious. Figures 15–17 show the impact of the percentage of malicious fog nodes in all fog nodes on energy consumption. Obviously, the increase of the proportion of malicious fog nodes and the increase of the number of fog nodes have the opposite effect on energy consumption.

6. Conclusions

In this work, we put forward a latency-aware energy-efficient continuous data-flow optimization strategy. This strategy is designed for continuous data flow applications in IoT-fog-edge computing scenarios. The most typical application of continuous data-flow is E-health Monitoring System. We made use of a novel lightweight anomaly detection strategy to get the confidence of the fog and MEC nodes. We used the confidence as the metric to evaluate the reliability of each nodes and use it to estimate the latencies in the energy-efficient continuous data-flow problem with latency constraints. We established a formal model and solved the problem using the block coordinate descend max-flow (BCDM) algorithm. The real-life datasets were used in the numerical study to verify the performance of the proposed strategies. Numerical results showed that the proposed strategies have good performance in all simulations.

In this paper, we only consider the latency property of data flow service. However, some other network attributes will also have a great impact on the overall performance of the system, such as frequency of messages, message rates, size, etc. We will combine these attributes with latencies for measuring the system performance in our future work. Although the main motivation scenario of the continuous data flow problem in this paper is E-health monitoring system, the continuous data flow problem can also be extended to more application scenarios, such as mobile social network [40], intelligent industrial monitoring system [41], etc. We will further consider the location of fog nodes and MEC nodes in continuous data flow problem [42]. Furthermore, we will conduct more simulations in an event-driven simulator [43], such as the YAFS [44], in our future work.

Author Contributions: Formal analysis, Y.L.; Funding acquisition, Y.L. and S.Q.; Methodology, Y.L. and W.L.; Resources, W.L.; Validation, S.Q.; Writing—original draft, Y.L.; Writing—review & editing, S.Q. All authors have read and agreed to the published version of the manuscript.

Funding: This work is funded by the National Natural Science Foundation of China (Grant No.61303043,61802030,61772087), the Hunan Provincial Education Science '12th Five-Year' planning subject (XJK015QJG001), and the Changsha Science project (K1705091), the International Cooperative Project for "Double First-Class", CSUST (No. 2018IC24).

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

References

1. Li, W.; Chen, Z.; Gao, X.; Liu, W.; Wang, J. Multi-model framework for indoor localization under mobile edge computing environment. *IEEE Internet Things J.* **2018**, *6*, 4844–4853. [CrossRef]
2. Lu, X.; Qian, X.; Li, X.; Miao, Q.; Peng, S. DMCM: A Data-adaptive Mutation Clustering Method to identify cancer-related mutation clusters. *Bioinformatics* **2018**, *35*, 389–397. [CrossRef] [PubMed]
3. Aazam, M.; Zeadally, S.; Harras, K.A. Deploying fog computing in industrial internet of things and industry 4.0. *IEEE Trans. Ind. Inform.* **2018**, *14*, 4674–4682. [CrossRef]
4. Pan, J.; Liu, Y.; Wang, J.; Hester, A. Key Enabling Technologies for Secure and Scalable Future Fog-IoT Architecture: A Survey. *arXiv* **2018**, arXiv:1806.06188.
5. Hu, Y.C.; Patel, M.; Sabella, D.; Sprecher, N.; Young, V. Mobile edge computing—A key technology towards 5G. *ETSI White Pap.* **2015**, *11*, 1–16.
6. Morris, I. *ETSI Drops "Mobile" from MEC*; Light Reading: New York, NY, USA, 2016.

7. Kaur, K.; Garg, S.; Aujla, G.S.; Kumar, N.; Rodrigues, J.J.; Guizani, M. Edge computing in the industrial internet of things environment: Software-defined-networks-based edge-cloud interplay. *IEEE Commun. Mag.* **2018**, *56*, 44–51. [CrossRef]
8. Pinto, S.; Gomes, T.; Pereira, J.; Cabral, J.; Tavares, A. IIoTEED: An enhanced, trusted execution environment for industrial IoT edge devices. *IEEE Internet Comput.* **2017**, *21*, 40–47. [CrossRef]
9. Mäkitalo, N.; Ometov, A.; Kannisto, J.; Andreev, S.; Koucheryavy, Y.; Mikkonen, T. Safe and secure execution at the network edge: A framework for coordinating cloud, fog, and edge. *IEEE Softw.* **2018**, *35*, 30–37. [CrossRef]
10. Guo, M.; Li, L.; Guan, Q. Energy-Efficient and Delay-Guaranteed Workload Allocation in IoT-Edge-Cloud Computing Systems. *IEEE Access* **2019**, *7*, 78685–78697. [CrossRef]
11. Zhang, L.; Wang, K.; Xuan, D.; Yang, K. Optimal task allocation in near-far computing enhanced C-RAN for wireless big data processing. *IEEE Wirel. Commun.* **2018**, *25*, 50–55. [CrossRef]
12. Chen, M.H.; Liang, B.; Dong, M. Joint offloading and resource allocation for computation and communication in mobile cloud with computing access point. In Proceedings of the IEEE INFOCOM 2017-IEEE Conference on Computer Communications, Atlanta, GA, USA, 1–4 May 2017; pp. 1–9.
13. Pereira, C.; Pinto, A.; Ferreira, D.; Aguiar, A. Experimental characterization of mobile iot application latency. *IEEE Internet Things J.* **2017**, *4*, 1082–1094. [CrossRef]
14. Santos, G.L.; Endo, P.T.; da Silva Lisboa, M.F.F.; da Silva, L.G.F.; Sadok, D.; Kelner, J.; Lynn, T. Analyzing the availability and performance of an e-health system integrated with edge, fog and cloud infrastructures. *J. Cloud Comput.* **2018**, *7*, 16–37. [CrossRef]
15. Farahani, B.; Firouzi, F.; Chang, V.; Badaroglu, M.; Constant, N.; Mankodiya, K. Towards fog-driven IoT eHealth: Promises and challenges of IoT in medicine and healthcare. *Future Gener. Comput. Syst.* **2018**, *78*, 659–676. [CrossRef]
16. Rahmani, A.M.; Gia, T.N.; Negash, B.; Anzanpour, A.; Azimi, I.; Jiang, M.; Liljeberg, P. Exploiting smart e-Health gateways at the edge of healthcare Internet-of-Things: A fog computing approach. *Future Gener. Comput. Syst.* **2018**, *78*, 641–658. [CrossRef]
17. Ibidunmoye, O.; Hernández-Rodriguez, F.; Elmroth, E. Performance anomaly detection and bottleneck identification. *ACM Comput. Surv. (CSUR)* **2015**, *48*, 4–40. [CrossRef]
18. Khalil, I.; Bagchi, S. Stealthy attacks in wireless ad hoc networks: Detection and countermeasure. *IEEE Trans. Mob. Comput.* **2010**, *10*, 1096–1112. [CrossRef]
19. Xie, K.; Li, X.; Wang, X.; Cao, J.; Xie, G.; Wen, J.; Zhang, D.; Qin, Z. On-line anomaly detection with high accuracy. *IEEE/ACM Trans. Netw.* **2018**, *26*, 1222–1235. [CrossRef]
20. Xie, K.; Li, X.; Wang, X.; Xie, G.; Wen, J.; Zhang, D. Graph based tensor recovery for accurate Internet anomaly detection. In Proceedings of the IEEE INFOCOM 2018-IEEE Conference on Computer Communications, Honolulu, HI, USA, 16–19 April 2018; pp. 1502–1510.
21. Xie, G.; Xie, K.; Huang, J.; Wang, X.; Chen, Y.; Wen, J. Fast low-rank matrix approximation with locality sensitive hashing for quick anomaly detection. In Proceedings of the IEEE INFOCOM 2017-IEEE Conference on Computer Communications, Atlanta, GA, USA, 1–4 May 2017; pp. 1–9.
22. He, S.; Xie, K.; Xie, K.; Xu, C.; Wang, J. Interference-Aware Multisource Transmission in Multiradio and Multichannel Wireless Network. *IEEE Syst. J.* **2019**, *13*, 2507–2518. [CrossRef]
23. Sohal, A.S.; Sandhu, R.; Sood, S.K.; Chang, V. A cybersecurity framework to identify malicious edge device in fog computing and cloud-of-things environments. *Comput. Secur.* **2018**, *74*, 340–354. [CrossRef]
24. Li, W.; Song, H.; Zeng, F. Policy-Based Secure and Trustworthy Sensing for Internet of Things in Smart Cities. *IEEE Internet Things J.* **2018**, *5*, 716–723. [CrossRef]
25. Brogi, A.; Forti, S.; Guerrero, C.; Lera, I. How to Place Your Apps in the Fog-State of the Art and Open Challenges. *arXiv* **2019**, arXiv:1901.05717.
26. Wu, Q.; Zhang, R. Common throughput maximization in UAV-enabled OFDMA systems with delay consideration. *IEEE Trans. Commun.* **2018**, *66*, 6614–6627. [CrossRef]
27. Wu, Q.; Zeng, Y.; Zhang, R. Joint trajectory and communication design for multi-UAV enabled wireless networks. *IEEE Trans. Wirel. Commun.* **2018**, *17*, 2109–2121. [CrossRef]
28. Sun, X.; Ansari, N. Edgeiot Mobile edge computing for the internet of things. *IEEE Commun. Mag.* **2016**, *54*, 22–29. [CrossRef]

29. He, Q.Q.; Yang, W.C.; Hu, Y.X. Accurate method to estimate EM radiation from a GSM base station. *Prog. Electromagn. Res.* **2014**, *34*, 19–27. [CrossRef]
30. Das, K.; Schneider, J.; Neill, D.B. Anomaly pattern detection in categorical datasets. In Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, NV, USA, 24–27 August 2008; pp. 169–176.
31. Dutta, S.; Nayek, P.; Bhattacharya, A. Neighbor-aware search for approximate labeled graph matching using the chi-square statistics. In Proceedings of the 26th International Conference on World Wide Web. International World Wide Web Conferences Steering Committee, Perth, Australia, 3–7 April 2017; pp. 1281–1290.
32. Wang, Y.T.; Bagrodia, R. ComSen: A detection system for identifying compromised nodes in wireless sensor networks. In Proceedings of the Sixth International Conference on Emerging Security Information, Systems and Technologies, Rome, Italy, 19–24 August 2012; pp. 148–156.
33. Kumar, S.; Dutta, K. Intrusion detection in mobile ad hoc networks: Techniques, systems, and future challenges. *Secur. Commun. Netw.* **2016**, *9*, 2484–2556. [CrossRef]
34. Kathiroli, R.; Arivudainambi, D. Election of Guard Nodes to Detect Stealthy Attack in MANET. In *Wireless Communications, Networking and Applications*; Springer: New Delhi, India, 2016; pp. 127–140.
35. Floudas, C.A. *Nonlinear and Mixed-Integer Programming-Fundamentals and Applications*; Oxford University Press: Oxford, UK, 1995; Volume 4, pp. 249–281.
36. Boyd, S.; Vandenberghe, L. *Convex Optimization*; Cambridge University Press: Cambridge, UK, 2004.
37. Grippo, L.; Sciandrone, M. On the convergence of the block nonlinear Gauss–Seidel method under convex constraints. *Oper. Res. Lett.* **2000**, *26*, 127–136. [CrossRef]
38. Li, W.; Song, H. ART: An attack-resistant trust management scheme for securing vehicular ad hoc networks. *IEEE Trans. Intell. Transp. Syst.* **2015**, *17*, 960–969. [CrossRef]
39. Barlacchi, G.; De Nadai, M.; Larcher, R.; Casella, A.; Chitic, C.; Torrisi, G.; Antonelli, F.; Vespignani, A.; Pentland, A.; Lepri, B. A multi-source dataset of urban life in the city of Milan and the Province of Trentino. *Sci. Data* **2015**, *2*, 150055. [CrossRef]
40. Zeng, F.; Yao, L.; Wu, B.; Li, W.; Meng, L. Dynamic human contact prediction based on naive Bayes algorithm in mobile social networks. *Softw. Pract. Exp.* **2019**. [CrossRef]
41. Li, W.; Xu, H.; Li, H.; Yang, Y.; Sharma, P.K.; Wang, J.; Singh, S. Complexity and Algorithms for Superposed Data Uploading Problem in Networks with Smart Devices. *IEEE Internet Things J.* **2019**. [CrossRef]
42. Zeng, F.; Ren, Y.; Deng, X.; Li, W. Cost-effective edge server placement in wireless metropolitan area networks. *Sensors* **2019**, *19*, 32. [CrossRef] [PubMed]
43. Perez, D.A.; Velasquez, K.; Curado, M.; Monteiro, E. A Comparative Analysis of Simulators for the Cloud to Fog Continuum. *Simul. Model. Pract. Theory* **2019**, 102029. [CrossRef]
44. Lera, I.; Guerrero, C.; Juiz, C. YAFS: A Simulator for IoT Scenarios in Fog Computing. *IEEE Access* **2019**, *7*, 91745–91758. [CrossRef]