The Reinforcement Learning Method in Deciding on Task Offloading and Energy Consumption Management at the Edge of IoT Network

Asghar Mohammadian[a], Houman Zarrabi[b,*], Sam Jabbehdari[c], Amir Masoud Rahmani[a]

[a]*Department of Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran,Iran*
[b]*IRAN Telecommunication Research Center (ITRC), Tehran, Iran*
[c]*Department of Computer Engineering, North Tehran Branch, Islamic Azad University,Tehran, Iran*

Abstract

In this paper, using edge processing in IoT network ,a method to decide on task offloading to edge devices and improve management of consuming energy in network of edge devices is introduced. First, by defining the problem of maximizing utility and then by decomposing it based on the status of task offloading from end devices to smart gateway with the lowest battery consumption and the possible lowest use of the communication bandwidth, independent optimal models are obtained and then combining. Then the general problem of maximizing the usefulness and increasing the lifetime of the end devices with restrictions of processing time and energy resources will be obtained. Due to the unknown environment of the problem and how the end devices and the edge of the network, an iterative reinforcement learning algorithm is used to generate the optimal answer in order to maximize the utility gain.The results show the existence of processing overhead and network load with increasing number of devices. The proposed method, while improving energy consumption in existence of a small number of devices in end of edge, reduces latency and increases processing speed and maximizes system performance compared to the central cloud system. The operating efficiency of the whole system is improved by 36% and the energy consumption at the edge of the network is optimized by 12.5%. It should be noted that, with the addition of a large number of end devices, our method outperforms similar works.

Keywords: Internet of Thing, Edge Computing, Task Offloading, Energy Consumption Management.

1. Introduction

With the development of IoT(Internet of Things) devices, energy consumption management has become essential. Data processing and computing methods are developing

*Corresponding author
Email addresses: Asgharmohamadian@gmail.com (Asghar Mohammadian), Houman.Zarabi@gmail.com (Houman Zarrabi), s_jabbehdari@iau-tnb.ac.ir (Sam Jabbehdari), rahmani@srbiau.ac.ir (Amir Masoud Rahmani)

every day. To increase the quality of service and reduce the volume of data traffic and delays as well as reducing energy consumption,edge processing and data exchange are required at the edge network. Because of the complexity and redundancy of data processing in each edge device and data transfer between devices and at different levels of the system, energy consumption is increased. Therefore, due to the limitations of end devices at the edge of IoT networks, energy consumption management at this level is essential[1]. So far, cloud computing methods have sometimes expanded cloud resources to the edge network. But, because of the size of the IoT network and the existence of heterogeneous devices at the edge of the system, this expansion could not be effective in energy management at the side of the net. [2]. Edge computing techniques can not only access the central cloud but also access the data and process offloading taken from the end devices and can exchange data [3]. Although some part of the data processing can be done in the cloud, based on the organization of edge computing, but processing management at the edge of the network is very important in terms of energy consumption [3].

IoT networks on edge consist of sensors, actuators, and the embedded processing unit and other devices available to storage and wireless communicated [2]. However, limitations such as size, reduce the ability of any IoT device's processing and energy resource. Furthermore, wireless transmission consumes energy, [4] and more energy is needed to improve service quality [5]. Therefore, to save energy on the edge of the IoT network, QoS(Quality of Service) must be optimized at the same time. Improving QoS, reduces transmission bandwidth, transmission delays and data packet loss. Therefore, the balance Established between QoS and limited resources can save energy.

In the edge computing, by considering edge servers near end-users [6], has provided a model in which network target services are deployed on edge servers to reduce network latency to provide better QoS. They have solved the problem of EUA(Edge User Allocating) service providers by maximizing the number of dedicated edge servers and computing resources employed on the edge servers while ensuring their network QoS. They have proposed an exploratory method for finding optimal solutions for large-scale cases that, while reducing delays and improving the quality of appropriate services, also improves energy consumption to some extent [6].

Examining the development of the IoT network, it is observed that the network environment is becoming increasingly complex, and the sources of communication and energy consumption are becoming increasingly scarce, which is a significant challenge for IoT networks [7]. As the number of wireless devices on IoT proliferates, more data is being obtained from the environment by network nodes. Therefore, the method of fast data forwarding with limited storage space of end devices and limited bandwidth is a big challenge for the current wireless network of IoT. Optimal energy consumption by end devices simultaneously with the above problems, adds to the importance of using a smart method to manage energy [8]. Being aware of the importance of QoS in the IoT network, developing algorithms and methods to find the right balance between computing at the edge of the system, QoS, and power efficiency to extend the life of the network and devices is essential.

Because end devices have batteries, they will have a shorter lifetime due to high energy consumption for data transmission and processing. The use of reinforcement learning methods in intermediate devices reduces the amount of offloading, and prevents excessive energy consumption in the end devices [9],[10]. Therefore, learning-based energy management methods such as Q-learning should be used at the edge of the network,[11]

to balance the proposed parameters of QoS, and increase the lifetime of the system by managing energy consumption.
Here, reinforcement learning method is proposed for energy management on IoT systems. This method uses optimal control measures in time-varying dynamic energy management systems. Of course, in these systems, despite the challenge of data transfer and bandwidth limitations, edge computing techniques are used yet[10]. Furthermore, in IoT-based energy management systems, edge computing can provide computational services on the edge of network near IoT devices and reduce the volume of the portable data using pre-processing methods [12]. Also, edge computing can enhance the limited capabilities of low-power devices by utilising computational demand and energy management[13]. Studies show that in energy consumption management in IoT systems the following parameters and methods are important:
QoS requirements in the IoT communications, selecting the location of data processing on the network and edge computing method and communication latency and method of control of the computing process between end-devices and network devices. In IoT devices, communication and computing solutions and network data transmission have an effect on the QoS provided. The high expectations of QoS with the precise performance of the communication network can affect the energy consumption of the devices at the network level. According to the complexity of network infrastructure and QoS requirements in resource management, managing energy consumption IoT devices focus on various aspects of optimization, such as minimizing latency, optimizing resources, and minimizing costs [14].
QoS features are essentially depending on the preferences and limitations of the wireless network edge, respectively, and since there is a surface QoS in each layer of the IoT model, the selection of the above requirements is done according to the network edge computing model [15]. The general QoS requirements of the system at the IoT edge level, are as follows. Of course, depending on the problem definition model and the intended application, some QoS requirements can be assumed to be fixed as below [16]; bandwidth, throughput, processing priority, scheduling policy, cache policy, quantum timing, frame rate, frame Size and resolution, response time, battery lifetime, cost and reliability.
Here, an algorithmic method of energy consumption management based on reinforcement learning is presented at the edge of the IoT network. Also, edge computing and task offloading methods are used to improve processing and transmission time at the edge. In addition, some QoS requirements are considered at edge of the network to manage energy usage.
This research continues with the following structure: In Section 2, related works with energy consumption management in the IoT are discussed. In Section 3 the IoT energy consumption management algorithm and it's requirements are described and in section 3.3 the energy consumption model at edge of IoT network and the proposed method in reinforcement learning and QoS requirements information are represented. Then, a simulation setup also simulation and model results are brought in Section 5. The results indicate the performance of the above methods. Conclusions of the discussion and future work are presented in Section 6.

2. Related work

In this part, a summary of the related works will be presented. Creating dynamic edge nodes and maintaining an edge network that they can load their task, are challenging[17]. Also, for efficient use of edge computing resources, energy consumption management and new resource management programs are needed for the architecture of the hybrid cloud/edge networks. There are several ways to manage energy consumption in IoT systems. For example, a state of the art algorithms and energy conservation approaches according to the energy consumption management provided by [18].

In [19], a powerful way with task offloading by use of cloud computing is presented. Online offloading is a difficult task and they use an algorithm that minimizes the time it takes to use the cloud service.

According to [20], the use of task offloading and resource allocation for MEC(Mobile Edge Computing) reduces the MEC delay and edge computing time. Finally, as stated in [21], the issue of energy optimization consists of three combined problem solving methods that are solved by non-linear programming method with a combination of the following: offloading decision, resource allocation, and over-clocking decision. Solving the above problems in MEC systems, the delay is reduced, and the energy consumption of the MEC system is optimized. [22] has proposed an average time calculation rate maximization algorithm that allows the joint allocation of radio and computation resources and the harvested energy can be used. They have powered by the wireless harvested energy signal to extend their device's lifetime in the IoT network. In any time block, the proposed algorithm decides about the optimal bandwidths, transmit powers, time allocations for computing on the device and task offloading, and the device local CPU(Central Processing Unit) frequencies. Finally, [22] also used a dynamic resource allocation technique for the optimized interchange between the energy utilization of the user device and the total operation delay.

In [23], the problem of energy efficiency optimization adjust QoS requirements according to the stability of the battery of devices with random optimization, which is challenging to solve due to connection time limitations, and that is one of the NP-hard issues. They have used the Lyapunov optimization, which uses current user modes and does not belong to system statistical information. Also, the issue of energy management for MEC systems has been examined. At the same time, MU(Mobile Units)s equipped with EH(Energy Harvesting) tools with QoS restrictions have been considered. The authors have developed a loading method based on Lyapunov's optimization process for energy exchange during execution. They have optimized energy consumption with formulating the optimization problem and edge computing capacity limitation and implementation delay time. Also, to maximize network efficiency by creating a balance on efficiency, they have designed a disturbed Lyapunov function that has been solved a knapsack problem per time slot for the optimal program. In addition, they also tried to maintain the Qos requirements in MEC.

In [24], Energy management and service quality in the vast IoT environment have been done with thousands of units consisted of objects and services. They have defined the conscious choice of QoS services to provide a combination of services as a difficult NP problem. In [24], ILP has been used to solve the problem of service selection. The issue of service selection has been developed to dominate the restriction of the ILP-based approach. Each of which, deals with a QoS feature. In the hybrid model, services selection has been formulated as a MMKP(Multidimensional Multichoice Knapsack Problem),

where the graph model describes the election as a MCSP(Multi-Constraint Shortest Path) problem. The main goal of the authors is to access better quality of services by minimizing energy utilization, ie maximizing the battery of the device. The primary purpose of [24] is to use the energy consumption specifications and QoS features and the user's priority in computing at the edge of the mobile objects network.
Furthermore, [25]has proposed a method based on workload composition and propagations to existing cloud management software. They have developed a novel technique to energy yield in virtuallized infrastructures, and in this way, they have increased energy yield. Their proposed method allowed the prediction of resources in virtual mode and freezes them if unused, to energy saving in the base infrastructures.
[26] has developed a game-theoretic method for achieving QoS-aware calculation offloading in a distributed network system. They investigated the multi-hop computation-offloading problem for the IoT-edge-cloud computing model and communication-routing issues to minimize each task's processing time and energy utilization. By proposing a scheme by the MDP(Markov Decision Making Process) for resource allocation and task scheduling in the edge in [27], the computing paradigm has examined the issue of energy efficiency of the IoT system. They have balanced the trade-off between energy consumption and QoS requirements. [28] has examined the capabilities of short-range wireless devices to connect to other power equipment such as a smart-phone as a GW(Gateway) for the Internet connection. The method used, is the M2M(Machine to Machine) method. The authors have also examined the quality requirements of QoS services for selecting the appropriate M2M gateway for terminal device connectivity. The application of different operating models presented in M2M gateway selection using the short-range wireless system, has also provided significant optimal results in terms of energy consumption.
While describing the trend of IoT edge computing in [29], its effect on the energy consumption of IoT devices has also been examined. Then, the overhead caused by the computing at the edge has been evaluated.
In [30], edge computing has been proposed as a promising way to increase capacity and resource management on IoT devices to support QoS and QoE(Quality of Experience) on the network and its devices. According to [31], the interaction between IoT end devices on edge, affects not only the QoS of system and latency but also energy consumption and battery lifetime of the devices. They have developed IoT network resource management strategies. They have challenged the management of processing and tasks in IoT and have studied the various applications of machine learning for data processing and managerial tasks. Also, They have classified and used modern applications of machine learning according to the type of data and their scope of application in IoT systems.
In [32], the optimal EMP(Energy Management Policy) in nodes of WSN(Wireless Sensor Network) has been obtained, which has the optimal power and the average optimal delay in the low SNR regime.They have modelled the energy conversion function used to transfer data to the central node as a linear function. They have also modelled the function of converting energy consumption to data transmission in nonlinear modes using specific exploration policies. In [33], the energy harvesting method has been used to optimize energy management to maximize network performance. They have used two reinforcement learning algorithms in the decision-making process at a discounted cost to solve the energy management and EH problems. Regardless of the form of conversion performance, they have proposed optimal energy management policies. [34]

has proposed a new MF(Matrix Factorization) matrix model with deep learning, which has completed the CNN(Convolutional Neural Network). They have used the learned hidden features of the neighbour node to select an active node or service features. The CNN learns the features of neighbour node, form a feature matrix, and evolves the features of the purpose service.The results of practical experiments in edge processing environment has verified that compared to similar methods, the proposed method in [34] could consistently achieve higher QoS results.

3. The energy consumption in edge of IoT network and it's requirements

3.1. Energy consumption model of IoT end device

There are several operational cycles during the performance of an IoT end device. These cycles starts by the system timer or according to the environmental motivations as follows; sensing / Actuating, Processing, Transmitting. Each of the above operational tasks consumes energy.

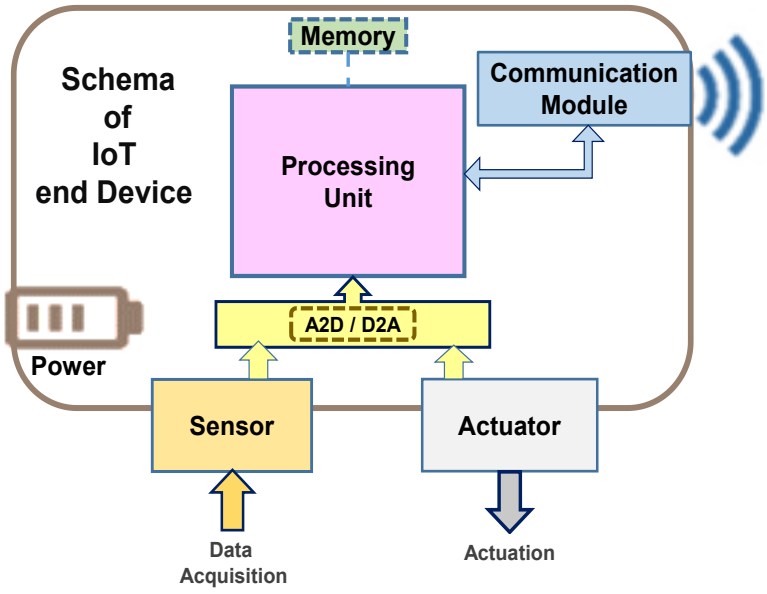

Figure 1: Schema of IoT end device model.

According to the schematic model of Fig. 1 which shows the essential components of these devices, their power consumption is spent in one of the areas of sensing or actuation, processing, communication, and the standby mode of the device[35]. If each IoT device is specified with an I_d, where $\mathbf{d} = \{1, 2, 3, \cdots, N\}$ and N are device

numbers, then the energy consumption of each I_d device at time t could calculate with using the power of each of the device operating processes as Eq. (1);

$$\mathbf{P_d}\left(\mathbf{t}\right) \triangleq S_d\left(t\right) + T_d\left(t\right) + C_d\left(t\right) + P_{std}\ . \tag{1}$$

Where, P_{std} is the required power of I_d in standby mode and $S_d\left(t\right)$ is the required power of the device to sense and store input data - $T_d\left(t\right)$ is the required power of the device to transfer data at moment t and $C_d\left(t\right)$ is the power consumption of I_d to process in device or process offloading to a higher level at moment t. By integrating the power consumption equations, the instantaneous energy consumption of each component of the device and its total energy consumption during its life cycle can be calculated as Eq. (2) and (3).
$E_{S_d} \triangleq \int_{t_{S_d}} S_d(t)\,\mathrm{d}t$, $E_{T_d} \triangleq \int_{t_{T_d}} T_d(t)\,\mathrm{d}t$, $E_{C_d} \triangleq \int_{t_{C_d}} C_d(t)\,\mathrm{d}t$.

$$\mathbf{E_d}\left(\mathbf{t}\right) \triangleq E_{S_d} + E_{T_d} + E_{C_d} + E_{std}\ . \tag{2}$$

$$\mathbf{E_{d_{Total}}} \triangleq \sum_{j} E_{S_{d_j}} + \sum_{k} E_{T_{d_k}} + \sum_{m} E_{C_{d_m}} + E_{std}\ . \tag{3}$$

Where, $E_{d_{Total}}$ is the total energy consumption of each end device of the IoT network, and $\mathbf{j} = \{1, 2, 3, \cdots, N_{S_d}\}$, $\mathbf{k} = \{1, 2, 3, \cdots, N_{T_d}\}$, $\mathbf{m} = \{1, 2, 3, \cdots, N_{C_d}\}$ and $E_{std} \triangleq P_{std} * T$. T is the time period of the total activity of each device.
Assuming the independence and approximate equality of energy consumption of each of the operating processes in the duty cycle of the IoT device Sensing /Actuating, Processing, Communication. If an independent x process occurs in the number of samples more and more, the average energy consumption $E(x)$ of the device for X process approximately will be as follows;

$$\mathbf{E}\left(\mathbf{x}\right) = (1/N) \cdot \sum_{n} X_n\ ,\ n = 1, 2, 3, ..., N\ . \tag{4}$$

According to the above assumption and Eq.(3) and Eq.(4), the average energy consumption of each device will be as follows;

$$\overline{E}_d = N_{S_d}.E_{S_d} + N_{T_d}.E_{T_d} + N_{C_d}.E_{C_d} + E_{std} \tag{5}$$

$$\Rightarrow E_d = \sum_{i} E_{d_i}\ ,\ i = 1, 2, 3, ..., N\ . \tag{6}$$

In Eq. (6), i is the process number of the device and $\overline{E}_{d_i}$ is the average energy consumption of a device with N independent processes, and E_d is the sum of the average energy consumed in IoT end devices.
Due to the autonomous operation of IoT end devices, the device's standby energy consumption is significant in the network. (Due to constant listening to the communication channel). For this reason, in energy consumption modeling in IoT systems, the energy consumption of Standby mode is also included [36]. Of course, in modeling, the device's energy consumption is not considered to perform its main tasks, and if necessary, we can model it in full detail. This modeling's primary purpose is only the

energy consumption of the device concerning the network and processing approach. According to Eq. (5) on the IoT end device's energy consumption during its operation period, the battery lifetime of each device in a local IoT network with N devices can be computed. Then, according to the remaining energy of the battery of each device, it is decided how to manage the energy consumption and how to perform the processing operation in the terminal device or load the processing to a higher-level device and also transfer the data.
If for each end device I_d the model vector is as follows;

$$\mathbf{I_d} = \left(x_d, R_d, B_d, e_d, U_d\left(x_{d_i}\right), S_d\left(x_{d_i}\right), T_d\left(r_{d_{ij}}\right), C_d\left(r_{d_{ij}}\right)\right) \tag{7}$$

where, x_d indicates the possible input data rate to I_d with a specified sampling frequency and a certain data resolution. Given that each IoT device offers its services at different M_d quality levels, and each level has a different input data ratio and different QoS, So;

$$\mathbf{x_d} = \left\{x_{d_i} | i \in [1, M_d]\right\} . \tag{8}$$

In 7, R_d represents a set of possible data transfer rates from I_d, which depends on x_{d_i} device input data rate and the IoT device offloading strategy. The level of offloading and transfer of processing determines how much of the device's input data is not processed by its processor system and is transmitted to the high-level gateway.
Q_d of an I_d demonstrates the different levels of device process offloading.
Also, $r_{d_{ij}}$ is the data transfer rate based on the level i and the input data rate of the device and the transfer level j and depends on the amount of data loaded from the common inputs to the gateway and the average output data of the processed device.

$$\mathbf{R_d} = \left\{r_{d_{ij}} | i \in [1, M_d]\ ,\ j \in [1, Q_d]\right\} . \tag{9}$$

B_d is the minimum remaining battery lifetime of the device and specifies the next charge time or battery replacement time.
Also, e_d is the residual energy of the battery of the IoT end device by which the device can continue to operate and it is related to the instantaneous energy consumption of each device;

$$\mathbf{e_d} = e_b\left(t\right) - E_d\left(t\right)\ . \tag{10}$$

$U_d\left(x_{d_i}\right)$ is a utility function that evaluates the quality of service provided for user when the end device is receiving data with x_{d_i} input rate.
$S_{d_t}\left(x_{d_i}\right)$ Specifies the required power of I_d at moment t for sensor / actuator and x_{d_i} input data storage at moment t.
$T_{d_t}\left(r_{d_{ij}}\right)$ is the required power of I_d in t for data transfer at $r_{d_{ij}}$ input data rate.
$C_{d_t}\left(r_{d_{ij}}\right)$ is the required power of I_d for data processing with x_{d_i} input rate below the transfer level j at t.
Therefore, in the Eq. (1) and the operational model of the I_d, the device's battery life storage to continue working from the moment t is calculated as follows;

$$\mathbf{b_{d_{i,j,t}}} = e_{d_{i,j,t}} / P_{d_{i,j,t}} \tag{11}$$

Where, $b_{d_{i,j,t}}$ is the remaining battery storage of I_d at moment t, when the device receives and processes input's data at the x_{d_i} rate or transmits data at the $r_{d_{ij}}$ data rate.

3.2. Local gateway energy consumption model according to its operational model

Gateways connect IoT end devices to higher levels of the network and internet, receiving data from IoT devices and sometimes processing them as needed, and then transmitting the final result to the edge processing or cloudlet level.
Due to the performance execution of gateway devices, the energy consumption dependence to those network load is low and linear [37]. Therefore, the change in energy consumption of their under load state is very small compared to their free state, so these devices also can be used in the processing of task offloading from the IoT end device.
According to the description of the network element using its power consumption in idle and active modes, the linear energy consumption of most network equipment relative to the network load, and the low energy consumption of the device in the network load mode [38], the power consumption of a gateway (P_{GW}) can be calculated using the power consumption of the idle state (P_{idle}) and energy required (E) for the maximum traffic load (R_{GW}) that can pass through that gateway, as follows:

$$\mathbf{P_{GW}}\left(\mathbf{t}\right) = P_{idle} + E.R_{GW}\left(t\right) \ . \tag{12}$$

Since the gateway does not generate significant traffic, and its power consumption in the idle mode can be 95% higher than its maximum power consumption for load transfer[37], so according to the Eq.(12) the energy consumption of a gateway during time T is as follows:

$$\mathbf{E_{GW}} = \int_0^T P_{GW}(t)\,\mathrm{d}t \ \approx \ P_{idle} \times T \ . \tag{13}$$

Data rate in a GW(Gateway) device is shown with R_{IoT} as follows:

$$\mathbf{R_{IoT}} = \sum_i R_{d_i} \ , for i = 1, 2, 3, \ldots, N \ . \tag{14}$$

Where, R_{d_i} is the data rate of IoT devices that are connected to a GW. If the background traffic of other devices connected to GW except the above I_d device is shown with R_{bgr}, then;

$$\mathbf{R_{bgr}} = \sum_s R_{vc_s} \ , for s = 1, 2, 3, \ldots, N \ . \tag{15}$$

Where, R_{vc_n} represents the traffic data rate of a unique, stand alone service and N_s is the total number of GW access services. Therefore, all R_{GW} traffic will be as follows;

$$\mathbf{R_{GW}}\left(\mathbf{t}\right) = R_{bgr}\left(t\right) + R_{IoT}\left(t\right) \ . \tag{16}$$

So the energy consumption in each gateways of the IoT network in T period base on data transferring ratio will be as below;

$$\mathbf{E_{GW_l}} = \sum_l P_{GW_l} \times T_l \ . \tag{17}$$

Where l is the number of levels under the network load in each GW set.
According to the issues, regarding the energy and power consumption of the gateway, each gateway in IoT network is modeled as follows:

$$\mathbf{GW} = \left(p_c\left(r_{d_{ij}}\right), R, P_C, P_{GW_l}\left(R\right)\right) \quad . \tag{18}$$

Where, $p_c\left(r_{d_{ij}}\right)$ indicates the ability to process the gateway according to the required processing needs of the I_d, which with the data ratio of $r_{d_{ij}}$, obtains the data from the end device at any time with offloading level of j to gateway. $R = R_{max}$ is the total bandwidth available to the gateway to receive data from the end device, P_C is the maximum ability of the gateway to process data, and $P_{GW_l}\left(R\right)$ is gateway power consumption at any time based on the data transfer rate of network devices for processing by the gateway in edge LAN location.Of course, in the above model, the effect of the environment and surrounding devices -such as interference and noise in data transmission - is not considered in the R and $T_d\left(r_{d_{ij}}\right)$ parameters, but can be modeled if needed.

3.3. Edge Network Energy Consumption Modeling

In modeling the energy consumption of network edge devices, a bottom-up approach is used to describe the energy consumption of each network edge element based on the number of devices in each network part and the model of each device [39].According to the schematic model presented for the lower part of the network edge in the Edge system in Fig. 2, energy consumption in the network is a function of the data traffic flow generated by the devices and the ability to process each element of the network to provide services to end applicants [40].Because IoT devices often generate bit streams, it is possible to split energy consumption between all network elements, regardless of the size, capacity, or power consumption of the device. Of course, this is made possible by the ability to delegate processing to higher-level devices.

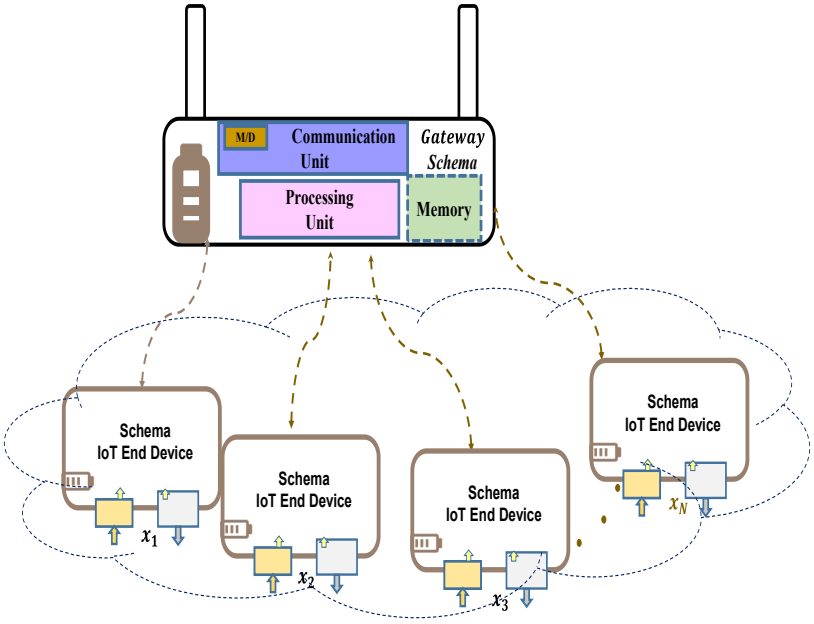

Figure 2: Operational model of a gateway device in the edge of IoT network.

Therefore, according to Eq. (12)and considering that the energy consumption in each time period in each device is $E = \int_{t_1}^{t_2} P(t)\,\mathrm{d}t$, it is possible to compute the required power consumption for each transmitted bit with transfer rate R, as follows:

$$\mathbf{P}\left(\mathbf{R}\right) = P_{idle} + E_{inc} \cdot R \quad . \tag{19}$$

Where, $E_{inc} \cdot R$ is the power consumption per bit transmitted with transfer rate R. Therefore, E_{inc}, which is the amount of energy consumption per bit transmitted at R ratio, and can be calculated as $(P_{max} - P_{idle})/R_{max}$. The device ability-based modeling approach divides the power consumption of the idle state of the network device and allocates it to the processing of offloading processes and services required by other devices. The required power of the device is based on the transmitted traffic flow using $P_{idle} \triangleq u \cdot R_{max}$, which u is the average amount of energy consumed in the processing service by the gateway. So, the average energy consumption transmitted per bit is assessed by a network element as follows;

$$\mathbf{E_{bit}} = (P_{idle}/(u \cdot R_{max})) + E_{inc} \quad . \tag{20}$$

Of course, many services have accessing the network elements in parallel and simultaneously. If the service provided by each network element to the A_{bit} data bits generated, received or transmitted consumes an additional amount of E_n energy, this additional

energy is computed by the following equation:

$$\mathbf{E_n} = E_{bit} \cdot A_{bit} \ . \tag{21}$$

Therefore, the energy consumption of the IoT devices in a network area for a GW is equal to:

$$\mathbf{E_{edge_L}} = \sum_i \left(E_{d_i} + E_{bit_{GW\,i}} \cdot A_{bit_i}\right) \ . \tag{22}$$

Where,E_{d_i} is the energy consumption of each IoT end device, and $E_{bit_{GW}}$ is the energy consumption of each local gateway per transfer data to higher levels, and A_{bit} is the number of bits transmitted.

3.3.1. Sharing a gateway device between multiple devices or multiple application services in terms of power and energy consumption based on data traffic transmission at the network edge

In a common gateway between several applicants, to determine the amount of energy consumption or the number of accesses to it, the part of the power of the device that is allocated to the desired service is considered. To achieve this important goal, it is practically impossible to determine the number of users or the number of elements or services that have shared this gateway. The simple way to model this gateway device is to use a traffic-based modeling method [41]. A part of the gateway power consumption (total power required for the transmission and idle mode)is allocated to the service with data traffic rate R_{srv}. In this case, the power consumption of the above service in time t will be as follows:

$$\mathbf{P_{srv}}\left(\mathbf{t}\right) = \left(P_{idle}/R_{IoT}\right) \cdot R_{srv}\left(t\right) + E_{inc} \cdot R_{srv}\left(t\right) \ . \tag{23}$$

Therefore, according to the above explanations, the energy consumption of each service from the end device in the gateway can be calculated by considering the integral of Eq. (23) in a time period of T;

$$\mathbf{E_{srv}}\left(\mathbf{t}\right) = P_{idle} \cdot \int_0^T \left(R_{srv}\left(t\right)/R_{IoT}\left(t\right)\right) \mathrm{d}t + E \cdot B_{srv}\left(T\right) \ . \tag{24}$$

Where, $B_{srv}\left(T\right)$ is the total amount of data bits transmitted by the services over time T. Therefore, the average energy consumption of each service in the time period of T can be calculated as follows:

$$\mathbf{P_{srv}}\left(\mathbf{T}\right) = \left(E_{srv}\left(T\right)/T\right) = P_{idle} \cdot \left(R_{srv}/R_{IoT}\right) + E \cdot R_{srv} \ . \tag{25}$$

Where, R_{srv}/R_{IoT} represents transmitted data ratio to total traffic transmitted to the device in period of T and we have:

$$0 \leq R_{srv}/R_{IoT} \leq 1 \ . \tag{26}$$

Considering the Eq. (26) values obtained from the above ratio, depend on the type of service and data flow speed, so the following three different service scenarios can be considered to examine the effect of the above average on the type and amount of energy consumption of each service:

Scenario 1: *In this scenario, a service provided with a fixed data transfer rate* $(R_{srv} \approx R_{constant})$ *is transferred to the network device (such as pacemaker monitoring),* thus;

$$\frac{R_{srv}}{R_{IoT}} \approx (1/R_{IoT}) \cdot R_{constant} \Rightarrow P_{srv} \approx (P_{idle} \cdot (1/R_{IoT}) + E) \cdot R_{constant} \quad (27)$$

Scenario 2: *In this scenario, a service is provided with a data rate that is approximately proportional to the total rate of data transmitted to the network device* $(R_{srv} \approx k \cdot R_{IoT})$ $, 0 \leq k \leq 1$;

$$\frac{R_{srv}}{R_{IoT}} \approx k \Rightarrow P_{srv} \approx P_{idle} \cdot k + E \cdot R_{srv} \quad (28)$$

Scenario 3: *In this scenario, a service is provided with a data rate almost negligible relative to the total transmitted rate to the network device* $\left(\frac{R_{srv}}{R_{IoT}} \approx 0\right)$;

$$P_{srv} \approx P_{idle} + E \cdot R_{srv} \; . \quad (29)$$

According to the above scenarios, the data flow and the total traffic rate transmitted to the network element, which is specified by R_{bgd}, can be defined as the sum of the input data rates of the services provided through the network device at any time:

$$\mathbf{R_{bgd}}\,(\mathbf{t}) = \sum_{l=1}^{n} R_{srv,l}\,(t)\,; l = 1, 2, \ldots, n \; . \quad (30)$$

Which, n is the number of services provided by the device for the data transferred to it. So the needed traffic for a processing service in the network device, in addition to $R_{bgd}\,(t)$,including data transmitted rate at moment t via the network element $(R_{IoT}\,(t))$ can also be calculated as follows;

$$\mathbf{R_{IoT}}\,(\mathbf{t}) = R_{IoT_{data}}\,(t) + R_{bgd_{srv}}\,(t) \Rightarrow R_{IoT}\,(t) = u\,(t) \cdot R_{max}, for 0 \leq u\,(t) \leq 1 \, . \quad (31)$$

According to Eq. (31), the power consumption of a common network device between several terminal devices or services with $R_{IoT_{data}}$ data rate is as follows:

$$\mathbf{P_{IoT}} = (P_{idle}/R_{max}) \cdot (R_{IoT_{data}}/u) + E \cdot R_{IoT_{data}} \; . \quad (32)$$

Hence a suitable metric for the average energy consumption per transmitted bit (idle mode + increased bit transfer) to a share network element with J/bit unit is as follows:

$$\mathbf{E_{BIT}} = ((P_{idle}/R_{max}) \cdot (1/u)) + ((P_{max} - P_{idle})/R_{max}) \, . \quad (33)$$

According to the description provided, IoT services that include sensors / actuators and a user who has access to a common gateway; the effective rate of gateway R_{effect} and, effective utility u_{effect} and gateway power consumption for each service j will be as follows:

$$\mathbf{R_{effect}}\,(\mathbf{t}) = \sum_{j=0}^{j} \sum_{k=1}^{k} R_{GW_{j,k}}\,(t) \; . \quad (34)$$

and,

$$\mathbf{P}\left(\mathbf{R_{GW_j}}\right) = \left((P_{idle} / \sum_{j=0}^{j} \sum_{k=1}^{k} R_{GW_{j,k}}) + ((P_{max} - P_{idle})/R_{max}) \right) + R_{GW_j} \ . \tag{35}$$

and, $u_{effect} = R_{effect}/R_{max}$.

4. Deciding on the process and task offloading and its impact on energy consumption management of the network

Deciding on the process offloading means which one of the tasks should be offloaded to a higher level and which one should be processed locally. Using the QoE based-utility function, the performance gain is measured over the system time period and the offloading energy consumption is compared with the local experience in the IoT device. Then, this matter is raised as a problem of maximum use of the system for the optimal using of communication, processing resources, and resource allocation problem in edge of IoT network in exchange for energy consumption. Due to the limited processing and energy resources at the network edge, a resource management approach is needed to save energy and reduce latency and ensure QoS at the network edge. Given that large data is obtained by end devices and in a continuous flow of data from the environment, to be processed in the cloud and the edge of network, the following three methods for processing at edge of the network are proposed:
1- Connection between the edge servers and the end devices for processing in the above servers.
2- Processing in the gateway devices and end devices.
3- Processing across the edge of the network on a large scale and locally on the network.
In case 1, several devices are located in the local edge server and transfer tasks to the edge server for processing. There are different types of tasks, including atomic tasks and tasks that can be shared between network devices for processing. In detachable tasks, the system has some independent data, which allows offloading data and processing on higher-level edge devices; Like health care systems. Most of the challenges in this area are due to scale. The ability to process at the edge of the network and load more tasks is related to the capabilities of edge devices, which often use stimulation mechanisms such as tit-for-tat mechanism. [42]. Due to the fact that edge servers may have other tasks, and also due to problems with uncertainty in the communication channel, there may be excessive or unnecessary delays in processing work on the edge server. Therefore, it is better to do the processing in more powerful devices in the middle of the network edge, such as local gates, and as a result, a trade-off between power consumption and latency is considered in edge processing due to the coverage area of the device.

4.1. Proposing a processing model at the edge of the IoT network

According to the proposed scenario, Fig. 3, the functional components include a local gateway with the possibility of performing limited processing and computing with a number of devices in the range of gateway services connected to the edge server. These devices can be mobile and it may has different resources including: processing and energy resources and so on. Each i device has $C.T_i$ computing capacity and it has the following features; input $D_i\ (bit)$, that includes program codes, input parameters,

system settings, and the number of cpu cycles needed to process tasks is C_i. Then, according to the stated cases, the ability of local calculations and processing time in the local gate device can be evaluated and modeled with the following assumptions.

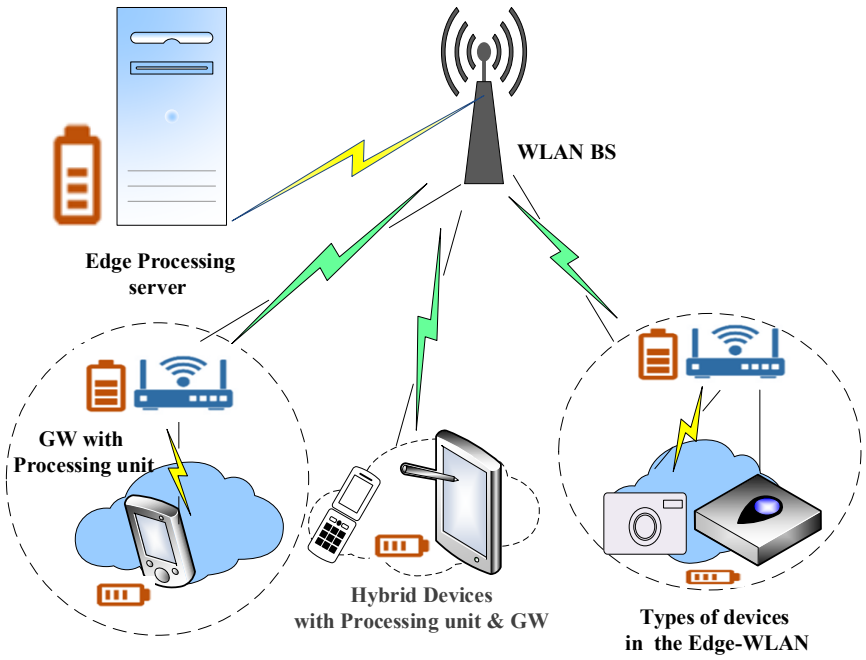

Figure 3: schematic of the functional components used in edge processing scenario of IoT.

Assuming the tasks are atomic and that the values related to the amount of input and transmitted data and the number of cpu cycles can be obtained using the application profile, any computational task can be processed locally on the device or by a processing center at the edge of the network. In local processing, ${F_i}^l$ is the ability of local computing of device i in terms of $Instruction/Sec.$ [43], [44];

$$\mathbf{T_i^l} = C_i/{F_i}^l \, . \tag{36}$$

According to [45], CPU power consumption is a linear function with a specific operating frequency, which is calculated as follows;

$$\mathbf{P_i^l} = \alpha.({F_i}^l)^\gamma \, . \tag{37}$$

Where, ${P_i}^l$ is the power consumption of each local cpu device and the values of α and γ are typically 10^-11 and 2, respectively [46]. Therefore, the energy consumption of each

device is based on its processing ability and power consumption(in joules) as follows [43], [44], [46];

$$\mathbf{E_i^l} = {P_i}^l.{T_i}^l = \alpha.({F_i}^l)^{(\gamma-1)}.C_i \ . \tag{38}$$

${F_i}^l$ and ${C_i}^l$ values are intrinsic features of each device and are obtained from the device profile. Computations in a device other than a local end device (eg. a gateway) usually consist of three steps:

1- The end device uploads the CT to the central station.

2- The device allocates Fi processing resources to the task processing and then processes it for the local device i.

3- The result of data processing is returned to the local device if needed, or it keeps it and sends it to a higher hierarchy in due time. (This storage or sending both consumes energy.)

Of course, in step 3, we do not consider the overhead caused by the obtained data. At the edge of the IoT network, remote computing improves processing and energy savings compared to local computing and processing with limited energy resource. However, transmission delays and processing priorities increase the overall system latency. According to the information about the energy consumption resulting from the transfer of data to the computing station, and according to the communication bandwidth and its operating model, the data rate of device and energy consumption will be as follows:

$$\mathbf{R_i}(\mathbf{P_i}) = w.\log_2\left(1 + ((P_i + h_i)/N_0)\right) . \tag{39}$$

Where,$w(Hz)$ is the bandwidth used to connect the local device to central station at the edge of the network. P_i is the power consumption for data transmission and N_0 is the noise power. Each IoT end device can adjust the transmitted data rate from zero to $w.\log_2\left(1 + ((P_0 + h_i)/N_0)\right)$ by controlling the power consumed to transfer data. Where P_0 is the power required for maximum data transfer.
The total time of remote processing for device i, is ${T_i}^l\left(f_i, P_i\right)$ and it consists of the following two parts;

$$\mathbf{T_i^l}\left(\mathbf{f_i}, \mathbf{P_i}\right) = {T_i}^t\left(P_i\right) + {T_i}^e\left(f_i\right) \ . \tag{40}$$

Where, ${T_i}^t\left(P_i\right)$ is transmission time at the channel, and ${T_i}^e\left(f_i\right)$ is remote execution time, that obtained based on size of the input data rate (D_i) and transmitted data rate.

$$\mathbf{T_i^t}\left(\mathbf{P_i}\right) = \frac{D_i}{w.\log_2\left(1 + a_i.P_i\right)} \ . \tag{41}$$

Where,$a_i = \frac{h_i}{N_0}$, and h_i indicates the increase / decrease of channel consumption by the local device I_d in data transfer to the Processing station at the edge of the network. ${T_i}^e\left(f_i\right)$ is also computed as follows:

$$\mathbf{T_i^e}\left(\mathbf{f_i}\right) = \frac{C_i}{f_i} \ . \tag{42}$$

Of course, in the above formulations, the transfer time of the channel and processing time in the processing device must not be infinite; that means: $f_i \neq 0$ and $P_i \neq 0$.

Power consumption of each device I_{d_i} for remote processing in the edge processing device is ${E_i}^r(j)$, which is assumed to be only the energy required for offloading, because by offloading the processing into the edge device, the energy to process of task in end device can be saved [46].

$$\mathbf{E_i}^{\mathbf{r}}(\mathbf{P_i}) = \frac{P_i}{\varsigma}.{T_i}^t(P_i) = \frac{P_i}{\varsigma}.\frac{D_i}{w.\log_2(1+a_i.P_i)} \ . \tag{43}$$

Where, ς is the increasing gain of the device power, which is achieved by saving on remote processing.

4.1.1. The formulation of Energy consumption management at the edge of IOT network based on the processing model at the edge

At the edge of the IoT network with the possibility of edge computing and processing, the quality of experience of the user i can be calculated by the task finalization time T_i and the amount of energy consumption E_i [47];

$$\mathbf{T_i} = s_i.{T_i}^r + (1-s_i).{T_i}^l \ . \tag{44}$$

$$\mathbf{E_i} = s_i.{E_i}^r + (1-s_i).{E_i}^l \ . \tag{45}$$

Where, $S \in \{0,1\}$ define the decision to offload to $C.T_i$, and if $s_i = 1$, Task is offloaded.

The improvement in latency of processing and energy consumption in the network edge processing device compared to the local computings in the IoT end devices is equal to $\frac{{T_i}^l - T_i}{{T_i}^l}$ and $\frac{{E_i}^l - E_i}{{E_i}^l}$, respectively. The settings of each applicant, during the task processing time and the total energy consumption for processing a task, are determined by ${\beta_i}^T, {\beta_i}^E \in [0,1]$, respectively, which are determined according to the battery lifetime of the device and its processing is needed until the end of the task. For example, a device i with a shorter battery lifetime will increase ${B_i}^E$ and decrease ${B_i}^T$, thus increasing battery lifetime by saving energy consumption.According to the above settings, the utility performance of each device i can be defined as follows,

$$v_i(s_i, f_i, P_i) = {\beta_i}^T \frac{{T_i}^l - T_i}{{T_i}^l} + {\beta_i}^E \frac{{E_i}^l - E_i}{{E_i}^l} = s_i.({\beta_i}^T \frac{{T_i}^l - {T_i}^r}{{T_i}^l} + {\beta_i}^E \frac{{E_i}^l - {E_i}^r}{{E_i}^l}). \tag{46}$$

As in the above model T_i and E_i are a function of the decision to offload from device i to the edge processing station (s_i). The gain v_i of each I_d measures the QoE improvement in offloading state relative to the local processing of computations on the same end device. Therefore, QoE could get improved in computing on the edge processing machine compared to the local processing on the end device. So, according to Eq. (44), if the decision is local processing on the I_d, $s_i = 0$, and the gain of offloading is $v_i = 0$. Also, due to bandwidth limitations and QoS restrictions and remote computing resources, overloading of tasks offloading for remote processing increases computation time and latency. If the remote computation latency time becomes too long, the computing utility gain becomes $v < 0$. Therefore, according to processing priorities, limited resources and service quality, it is possible to prioritize the loading of different users. In this case, the priority of each applicant for task offloading is determined with $\rho_i \in [0,1]$ and the utility

gain of the whole system for the available edge applicants will be $\sum_{i=1}^{|k|} \rho_i.v_i(s_i, f_i, P_i)$. Priority decisions are made by an intelligent algorithm according to QoS parameters and resource limitations and the amount of processes offloaded in the queue. Whatever ρ_i is close to 1, the more applicants will be in the queue priority for offloading and processing in the central local device. The above relation, while calculating the utility gain of all local IoT devices, also shows the tendensy of applicants to provide energy, computing, and processing resources. Therefore, sharing remote resources to offload the task becomes a maximized utility gain issue, which is formulated as follows [47], [44], [48];

$$\max_{s,f,P} \sum_{i=1}^{|k|} \rho_i.v_i(s_i, f_i, P_i) \tag{47}$$

Where,s, f, P are decision vectors for offloading. In which s Shows the allocation of processing/computing resources to the local I_d, f is the ability to data transferring and process computing from I_d to the edge computing device, and if the condition $f > 0$ is met, indicates the allocation of computing resources to all applicants of IoT end devices in the system. P is the power consumption of I_d in data or process offloading, computing and processing, which should be always positive and should not exceed P_0, that P_0 is the maximum power for data transferring. k is the number of local I_d in the edge group of IoT network. Due to the limitations of the system, the total computational and processing resources allocated by the central device to its local end devices should not exceed the maximum processing power of the computing center (f_0 with instruction per second). It should be noted that the maximum number of N devices are allowed to simultaneously transfer data to the edge computing device, that $N = \frac{B}{W}$.
The decision to offload processing is related to the optimization of network edge communication resources and computing and energy resources. In addition, since the above decision has an effect on the performance of the device and changes its energy consumption, so due to the continuity of changes in energy resources and computation, the above mixed problem can be formulated and solved by mixed nonlinear programming[49]. To solve the problem, first it will break down into the following dependency problems:

1- Joint optimization of energy resources, computing and communication for a special offloading intention.

2- Optimizing the decision of tasks offloading based on the result of optimizing energy, computational and communication resources.

So; Eq (48) is obtained;

$$\max_{s} v(s). \tag{48}$$

However, in the above state, the maximum N simultaneous transmitter device, previously described, must be considered. Therefore $v(s)$ is the maximum utility gain of system for a particular task offloading decision, when computing, energy and communication resources can be optimized simultaneously together. So, in the latter case the problem is as follows;

$$v(s) = \max_{f,p} \sum_{i \in S} \rho_i.v_i(s_i, f_i, p_i). \tag{49}$$

Where, $0 < P_i \leqslant P_0$, and $f_i > 0$, and $\sum_{i \in S} f_i \leqslant f_0$. According to the problem raised in Eq. (49), joint optimization of communication, computational and energy resources with the possibility of deciding on special offloading can be written as follows. Computational resources f in both of the above objectives will be separated according to the mentioned limitations. Therefore, problem in Eq. (49) as described earlier is solved independently with optimizing computational and communication resources and using remote resources.

4.1.2. Joint optimization of energy, computing and communication resources in edge of IoT with a bottom-up hierarchical view

According to the problem posed in Eq. (49), the problem of joint optimization of energy resources and computing to decide on processing or offloading can be solved by replacing Eq. (46) and according to the process offloading conditions, the power consumption of each end device to transfer data. Therefore, regarding the shared use of resources in the network edge processing station by the final devices, base problem Eq. (49) can be rewritten as follows;

$$\max_{f,p}(\sum_{i \in S} \rho_i.(\beta_i{}^T + \beta_i{}^E) - \sum_{i \in S} \rho_i.(\frac{\beta_i{}^T.T_i{}^r}{T_i{}^l} + \frac{\beta_i{}^E.E_i{}^r}{E_i{}^l})). \tag{50}$$

Pursuant to the analysis of the problem, according to Eq. (50), it is enough to maximize the utility gain and optimization at the mentioned problem, and minimizethe second part of the relation in Eq. (50). That is, during the completion time of the remote processing related to the device i, the edge processing station minimized the consumption of the above resources, in order to provide the requirements for the quality of operations and the quality of services.

$$\min_{f,P} \sum_{i \in S} \rho_i.(\frac{\beta_i{}^T.T_i{}^r}{T_i{}^l} + \frac{\beta_i{}^E.E_i{}^r}{E_i{}^l}). \tag{51}$$

First, the Eq.(51) is solved by inserting parametric values. Then each of the energy, communication and computing sources in offloading from the end device to the edge processing station are optimized. That is, each end device uses it's resources to minimize the following.

$$\min_{P_i} f(P_i)\,; 0 < P_i \leq P_0. \tag{52}$$

$$\min_{f_i} g(f_i)\,; for\ each\ i \in S, f_i > 0, \sum_{i \in S} f_i \leqslant f_0. \tag{53}$$

Considering the solution of the problem raised in Eq. (50) and the subproblem proposed in Eq. (51) and their sufferance conditions, the optimization of the computational resource are performed independently in the coverage radius of the processing station by solving Eq.(52) the optimization of the energy resource and by solving Eq.(53). Considering the discontinuity of the cases presented in the independent solution, while combining the decomposed problems Eq.(51), Eq. (52) and Eq.(53)and considering the general problem in Eq. (50), finally the basic problem is obtained as follows and it is

resolved using the integer nonlinear programming method[49].

$$\max_{S} \sum_{i \in S} \rho_i.({\beta_i}^T + {\beta_i}^E) - \min_{P} \sum_{i \in S} f(P_i) - \min_{f_i} g(f_i). \tag{54}$$

By solving the problem Eq.(54) with all the conditions presented, non-unique answers were obtained for each of the final devices within the coverage radius of the remote processing device, which can replace any of the answers of the maximum gain model. Given that the above problem is Np-Hard issue and can solved independently in data transfer and loading from device i to remote processing device, so due to the limited number of devices covered by the processing station and also because of the bandwidth limitation in the communication channel, this problem could be solved and exit the Np-Hard mode and has a unique and appropriate response at any time to decide whether to perform task offloading or local processing. Considering the problem of the whole system at the edge of the IoT LAN and the possibility of remote processing in each of the GWs, the effect of adding each new end device to the processing edge set (A), turns the problem into a new problem. Therefore, decisions on process transfer or local processing capability should be reconsidered in the system edge operating matrix;
Adding any new device to the network edge processing pattern can increase the overall utility gain. So with the addition of a new device to the edge of the network in the coverage area of each GW, the Eq. (55) will be obtained;

$$\mathbf{G_i}(\mathbf{V}(\mathbf{A})) = (A \bigcup \{i\}) - V(A) = \rho_i.({\beta_i}^T + {\beta_i}^E) - (G(i) + G(i|A)). \tag{55}$$

Where, $G(i)$ and $G(i|A)$ represent the utility gain of all fixed and variable parts of the network edge, which are computed in each new solution of the problem. $G(i)$ does not relate to the set of offloading decisions, but $G(i|A)$ increases uniformly by decision to offload set of A, and according this fact the problem is NP-hard stems. Of course, as mentioned, in real systems, due to the limitations of the number of local devices on the edge of the network and also the limitation of the bandwidth of communication system, the problem can be managed and solved. End devices may have different capabilities; In some it may be better to process locally, and in others it may be better to offload the process into a remote processor and GW. Of course, this issue, in addition to the capabilities of the device, also depends on the conditions of the transmission channel. However, the decision algorithm makes the relevant decision according to the device profile and service quality and adjusts the total utlity. If $max\ G_i(V(A)) \leq 0$ then local processing will be performed in each newly added device (applicant) . If $min\ G_i(V(A)) \geq 0$ then, the decision output will select the offloading mode from the device i to the remote processing device. In each of the selections and at each stage, all the parameters of the resources, energy and power consumption of all devices and the processing and remaining energy of the batteries of the devices are updated and transferred to the device profile and management program. If the primary applicants require more bandwidth than the available bandwidth ($|s| > N$), then by eliminating the minimum number of applicants, the desired bandwidth can be computed exploratory and directly. Applicants in this exploratory search will be selected to maximize $V(s)$ faster, and care must be taken to make $G_i(V_i(s)) > 0$ in each selection, in which case $S \bigcup \{i\}$ is also inseparable. Every time after updating the parameters, the output of the decision is provided to the central station so that it can choose the best applicant for

process offloading that has the optimal consumption of energy and other resources. In fact, in choosing the most suitable applicant in process offloading from the end device to the remote processing device (gateway), the main issue is to maximize the total utility of edge set. For this purpose, taking into account the processing priority of each end device in data/processing offloading (ρ_i), the ability of each device in offloading or processing (f_i) due to the $max\ G_i(V(A))$,device transfer ratio (r_i) and power consumption of each device to offload processing to the gateway at the edge of network (P_i), the following utility gain model can be formulated to manage consuming the energy of each end device during operation in the examined network edge set;

$$\max_{i,f}(\sum_{i\in S}\sum_{f}(u_{if}\times\rho_{if}). \tag{56}$$

$$u_{if}=u_i.(x_{i_f})$$
$$P_{if}=p_i.(r_{i_f})$$
$$\forall i\in S,\forall f\in i,\sum_{f\in i}\rho_{if}=1$$
$$\sum_i\sum_f(r_{i_f}\times\rho_{i_f})\leq R$$
$$\sum_i\sum_f(p_{i_f}\times\rho_{i_f})\leq P\ .$$

The main goal is to minimize processing and transfer latency and reduce energy consumption for IoT edge processing operations. To determine the optimal location for task processing, in order to reduce energy consumption, the following off-policy reinforcement learning and an efficient resource allocation mechanism is applied to increase the use of EC servers due to limited energy consumption and comparative resources compared in edge. In the proposed method as calculated in the previous sections; In order to solve the problem of resource management in edge, based on parameters derived from the environment such as data size, bandwidth, processing load to GW and edge servers, signal strength and power consumption, the necessary decisions are made and the learning process is performed.

4.2. Reinforcement learning process in energy consumption management by considering IoT edge processing

Reinforcement learning(RL) is one of the most basic methods of machine learning for better decision making in environments with uncertain characteristics. In this method, while the agent reacts to the environment state, the agent depending on the environment, learns type of function in next steps and performs the learned action in the next steps, and in return for doing so, receives the rewards and changes the state of the environment. This learning process is repeated until the total aggregation rewards received from the environment are enlarged and maximized[50]. The learned operation policy includes episodes in which the process of offloading, processing, and transferring data is repeated each time the devices are turned on until off. In addition, each time the reward for adopting the optimal policy of energy consumption and less use of network resources, such as high data transfer rate while short-term use of network bandwidth, is recieved

to improve service quality parameters.It is assumed the environment is demonstrated by a Markov decision-making method [51]in which the transfer of the operating state depends only on the current state and the action that takes place in the current state, so it does not depend on previous states and actions.
Given that Q-learning is a method based on reinforcement-learning in decision-making and management issues, the general process of $Q(s_t, a_t)$ learning is that Q-value is reached by calculating the pair of status and operation of a_t and s_t in a discrete time. Q-value calculates and updates the maximum reward using the Belman equation as follows;

$$Q^*{}_{v^*}(s_t, a_t) = r(s_t, a_t) + \gamma \cdot maxQ(s_{t+1}, a_{t+1}). \tag{57}$$

In Eq. (57), based on the ideal approach v^*, the ideal Q-value of $Q^*{}_{v^*}$ is calculated by adding the current $r(s_t, a_t)$ bonuses and the maximum future bonus of γ, and $Q(s_{t+1}, a_{t+1})$. $\gamma \in [0, 1]$ is a coefficient that shows the relative significance of future rewards. As γ coefficient decreases, the network processing agent becomes divergent and weak, while it increasingly focuses on the current reward.(Processing on the device) The larger γ, converges the factor and foretells future rewards. At each step of adding a new applicant, γ is randomly selected between [0,1]. But it is not configurable by the system designer. It should be noted that for greater convergence, γ must be selected larger than ρ_i at each step, so $\gamma \in [0, 1]$. At any time with the Q-value update, the new $Q(s_t, a_t)$ pair is stored in the energy status-process table, which includes the remaining energy mode of the device and the selection of the type of processing in the device or network, according to the described parameters based on maximum reward. The type of chosen processing in accordance to the values of the Q-table is up to date at any time primarily based on the Bellman equation as follows.

$$Q(s_t, a_t) \leftarrow (1 - \theta) \cdot Q(s_t, a_t) + \theta \cdot [r(s_t, a_t) + \gamma \cdot maxQ(s_{t+1}, a_{t+1})]. \tag{58}$$

In Eq. (58), $\theta \in [0, 1]$ suggests the quantity of getting to know that determines how properly the new Q-value overcomes the previous one. Whereas, with $\theta = 0$, the agent does not learn anything and only uses the previous Q-value without exploring the Q-learning process, and with $\theta = 1$, using only the current reward and maximum reward with the future coefficient, make the agent status updates. Same as the γ election, a collusion between finding and utilization could be defined using the $\theta \in [0, 1]$ value. By updating the $Q(s_t, a_t)$ in the next few iterations and using the Eq. (58), the value of Q becomes increasingly large, giving the optimal mode of processing source selection, energy consumption, and task offloading in the network device resource management matrix for maximizing the utility gain. V^* policy with maximum Q-value is as follows [48];

$$v^* = arg\ (max\, Q(s_t, a_t)). \tag{59}$$

This learning is an off-policy method in which the agent acts on the basis of a previous policy and decides on the current state of the system. In RL the efficiency of resource allocation and cloud computing and load calculation are stabilized [52]. The v^* policy estimates a status of the next state of action (s, a) based on the previous state of an action. To do this, the time difference TD is applied to each rule in each time period to allow the agent to move from one mode-action pair to another. Traditional reinforcement learning models for computation offloading have been used, for example in [53] an RL method for complex video games has been used and compared with other RL approaches. The

results obtained from the above article show that the Q-learning and SARSA-learning methods work better than other algorithms because they are the most rewarding. In this study, the Markov decision is used in RL to appropriately increase the reward in TRAINING TASK of a factor with the environment [54], so the future reward at time t is defined as follows;

$$R_t = \sum_{k=0}^{T} \alpha^k . r_{t+k+1} \tag{60}$$

Where $\alpha \in (0, 1]$ is a mitigation factor and r_t is a reward for a action at t When the agent in case s and time t acts a under policy v, it is shown as $Q^v(s, a)$. Where θ and γ is the expected reward and v is the policy function for the action A_t. The goal of the training is to get the maximum reward for optimal Q^v action and mode. The Q-Learning method is used here, because it has been shown in [55] that this method can provide an optimal path for the centralized offloading of tasks to the edge of the cloud. That is, the value of the next action is determined by the value of the current state (S_t) and the current action (a_t). The training is done with a multiple vector (s, a, r, Q^*) that is regularly updated at each stage. In this paper, the reinforcement learning process is used to a GW device to obtain the optimal processing performance program for a set of devices covered by a GW at the network edge and to select the best energy consumption mode according to the processing approach at the edge of network, and then the same output as the optimal policy will be transferred to other GWs at the network edge level. In this case, the energy consumption at the level of the whole edge network is optimized according to the processing approach at the edge of the network and the quality of service requirements of the whole system.

According to Fig. 3, the system first performs the process offloading to GW. If the task is large, it uses an edge server, and assuming that several edge servers and operating structures are adjacent to the IoT network edge, as in Fig. 3, it uses an adjacent edge server for processing if the edge server is busy. According to the explanations and the selected optimal policy, the iterated reinforcement method to perform an optimal solution to this problem, needs to obtain the maximum interest expressed and thats optimal policy for maximum energy use in processing and transfer of tasks and data rates at the edge. Assuming the maximization of energy efficiency and processing at the network edge in the previous sections, the value iteration method can be used in table Q to reinforce learning as follows;

Algorithm Q-RL for IoT Edge
Input: Number of MDs N and task size Dn
Output: efficient offloading decision, cost reduction and bandwidth allocation
Initialize the network parameters with upload and download bandwidth,
and processing cycle number
Initialize the number of iterations (episodes), let **I = 100**
for iteration **I < 1,2,3,. . . ., I** do
Select **"Action"** randomly.
Compute **"Current State"** according to formula No. 3
if**"Current state"< (St + 1)** then
Set **rt = 1**
else if **St> St+1** then
Set **rt = −1;**
else
Set **rt = 0;**
end if
Obtain reward rt and next state **St+1** after execution of at.
Set this as **(St, at,rt, St+1).**
Compute the **Q-value yt** from the target deep **QL yt = rt+1 + γ QSt+1, at+1**
Execute the algorithm of gradient descent to reduce
(yt − q (st+1, at+1); α)^2
Update **q-value: Q∗(s, a)**
end for.

Figure 4: QR-L Iteration Algorithm for max Value and Updates of Parameters in whole System

The above iteration algorithm, in Fig. 4 first initializes the function V(s) to the desired initial values (zero here) and then updates it with the value of the last Q obtained as it moves each step forward, in which at the same time; the values of data rate, bandwidth, power consumption, discharged processing and local processing time of the devices and latency in all operational matrices of each network edge processing device in the simulator are also updated (the same process can be transferred to the real device). After a number of iterations, the value v converges to the desired value of V^*, while the difference between the last two iterations is less than the very small value. In this way, the corresponding policy is related to the maximum policy (optimal pair of state and action). EDs are connected to the IoT edge network for processing at or near the edge of the network and to compute the amount of process offloading. An edge computing controller controls Multiple computing in each area. Because the components of the Edge IoT network have higher storage and processing capabilities than the ED and also have more energy, and the same resources are more and more in the remote cloud, so if resources beyond the edge are needed, this request is transmitted to the cloud via the EC controller. Assuming that there is more than one server and edge area adjacent to each other at the edge of the IoT network, as shown in Fig. 3, there are several APs, GW in each area, assuming the multiplicity of edge areas. There are several edge servers, and the number n is ED $(n = 1, 2, 3, ...)$. An ED can be connected to EC via Ap and GW. Depending on parameters such as response time, latency, and energy consumption,

EDs must find and select an effective location on the IoT edge network for offloading. As mentioned, the offloading location is taught by EDs based on γ and θ coefficient, and based on the decision to which part of edge IoT the offloading should be done. The decision to offload processing based on bandwidth and the effect of the data rate is taken and at each step the amount of computational delay is described as previously described. The amount of computational delay is also calculated as described earlier. In offloading on edge of IoT, the x_n coefficient is chosen for the decision; As shown $x_n = 0$ task transfers to GW and $x_n = 1$ task transfers to the nearest edge server, $x_n = 2$ transfers processing to the adjacent edge server, and $x_n = 3$ transfers processing to the remote tool. Therefore, based on the specified problem form and specific parameters such as the amount of edge server offload, latency and energy consumption, the decision is made at the selected processing location.

5. Simulation of Energy consumption managements in IOT and the results of performance

According to the cloud services modeled in Cloud-sim tool and considering the wireless access of the edge devices of the network to the local gateway and from there, to the edge processing elements and assuming the constant delay in the proposed model with end devices of being able to communicate, edge elements with the IoT network were modeled and simulated by this simulator. Assuming that the type and operating model of the network and end devices are fixed, the number of devices and access points to the network at the beginning of the simulation and according to the energy consumption model of the devices described earlier, the simulation and initial settings are performed. According to the proposed reinforcement learning method and the processing selection algorithm at the network edge, the amount of transmission delay and network edge traffic is computed based on the decision to offload / load the processing and data by the link. In this modeling, based on Cloud-sim and based on the model simulated in [56], the operational architecture of the simulation model has been developed. Then, the modeling data was analyzed in 1600 devices and compared with the architecture of the central cloud system. In the simulation model, the network is modeled based on a single-server queue module in the cloud model. Below the edge branch of this network is a Cloudlet server and finally gateways modules with the ability to process and data transfer to server. At the lowest layer of the network below the set of gateways have been modeled the end devices. Here, the processing system of each device is simulated taking into account the general model of energy consumption. There is no limit for the number of tasks submitted in terms of data transfer and processing tasks. The only noteworthy point is the bandwidth limit in simultaneous transmission of devices. However, due to the fact that at the edge level, tasks are performed in a very short time of a few milliseconds and in parallel on a virtual machine, so a special class has been used to simulate the edge processing model and decide about process offloading based on hypothetical Cloud-Sim tool model Fig. 5 shows the architecture of the task processing model at the edge of the network in simulation system [57]. In the emulator written based on the Cloud Sim core, an RL tool with a reward function is added, the main purpose of this function is to use the assignment of processing to reduce the processing delay of tasks. This primarily determines the response time while determining the processing capability of the EC controller and selects the appropriate position to assign the processing to the simulator. In the written tool, the possible actions

of RL are as follows;
Al, is a local processing.
AN, Assigns processing to the nearest edge server.
Aa, Assigns processing to the nearest adjacent edge server.
AR, Assigns processing to remote cloud.
ANA, transfers assigned processing from the nearest edge-to-edge server.
AAN, Transfers the assigned processing from an adjacent edge server to the nearest edge server.
ANR, Transfers assigned processing from the nearest edge server to the remote cloud.
ARN, transfers delegated processing from the remote cloud to the nearest edge server.
AAR, Transfers delegated processing from a remote cloud to an adjacent edge.
AGR, Assigns local processing to GW to obtain the result.
So the operating actions are generally specified as $A(t) = \{A_1(t), A_2(t), A_3(t), ..., A_n(t)\}$ in the simulator at any given time. As A_n represents the n'th decision and selection process offloading in the simulator. S mode is also the learning factor for processing resources at the edge of the network as follows;
Processing mode - Memory mode - Network bandwidth mode, the model of each of which is expressed in the previous complete sections, and therefore, the current state of the system can be computed from solving the problem model formulated at any time.
Of course in this system, a particular learning agent has no information about the general state of the nearest edges, the agent only has information about its local state. Collaboration and communication between agents and network element status information to delegate task processing depends on the location of the appropriate agent in the IoT network.

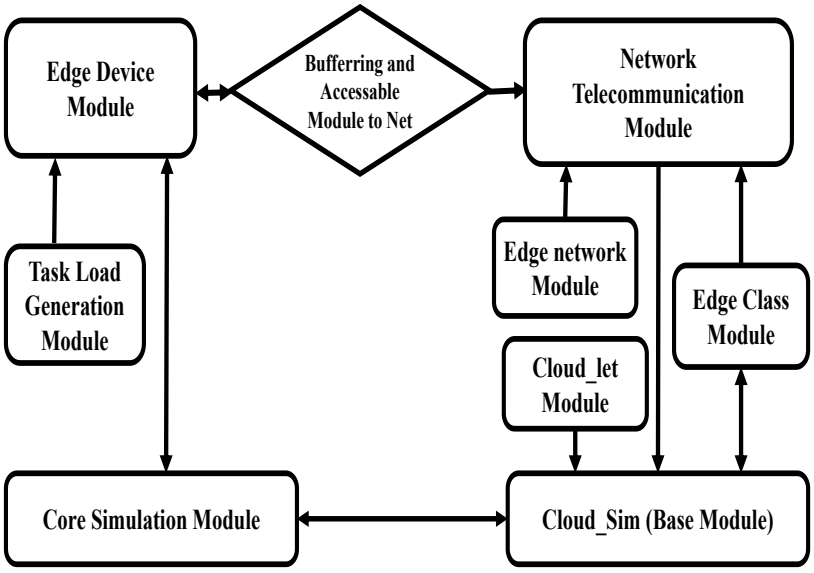

Figure 5: Architecture of the task processing model at the edge of the network in simulation system.

The architectural model used has two layers, the offloading ratio to the local edge server is 10%. Conditions are set to minimize offloading to the edge server for initial processing to reduce data latency and traffic due to data transfer at the edge of the network while maintaining data quality at GW, and help to maximize utility at the edge of the network. Due to the initial processing in GW and sending information to the Cloudlet server in due time, the type of end devices at the edge is not important. The model used, a data sending and processing model of the end device which has a separately processing feature. Because the model is simulated closer to the actual model, it uses the On/Off model to send process and data from end devices to the GW. The second assumption is that the sending model has a Poisson distribution and a sending and detection time of 3 seconds. There is also an idle-time and an active-time on each period of end device. Given that CloudSim focuses on the principles of processing operation, so the mobility of the edge devices of network is not considered in this framework, and only being within the coverage radius of the network access point is sufficient. The hash algorithm is used each time the status of existing devices is updated. Any GW device can use Wi-Fi and LTE. For simplicity, we have assumed the communication media at the edge of the network as Wi-Fi and the quality model of communication as fixed. The simulation parameters are according to the Table 1.

Table 1: Simulation parameters.

Parameter	Value
Poisson Inter arrival Time of Tasks (second)	3
Number of iterations	1000
Number of Device	1600
Number of Place Type	3
Probability of selecting a place type	1/3
Active/Idle period of the user (second)	15/3
Number of Edge/cloudlet Server per Place	1
CPU Utilization of task processing in Edge Server/Cloud	10%
Probability of Offloading to Cloud	10%
Probability of Offloading to GW	95%
Average Data Size for Upload/Download (KB)	1000/10
Average Task Size (Million Instruction)	1000
WAN/WLAN Bandwidth (Mbps)	25/250
WAN Propagation Delay (msec.)	100
VM Processor Speed (MIPS) per Edge Server/Cloud	1000/10000

5.1. Simulation Results

If the amount of use of the virtual machine to accept the offloading tasks are high or the network bandwidth is not enough to transfer the input/output of the tasks in a reasonable time, the tasks processing cannot be completed. Therefore, a task may not be fully processed due to lack of computational resources or lack of network and energy resources, and the service program may be facing a resource shortage. Here, to simplify the complexity of the simulations and according to explanation of the problem modeling, it is assumed that any task of device has a single processing at each offloading stage, independent of the type of activity of the end device. Processing of tasks/service delivery may fail due to mobility and departure from the network access point coverage area of real systems. However, due to the fact that this case covers a very small part of the operation, and due to the architecture of the simulation tool that focuses on processing, the assumption of mobility is ignored. The classification of devices and service inputs has been omitted because only the type of tasks is important.Another important point in the process of simulating and checking the outputs is the step-by-step addition of IoT end devices as a categorized procedure. In the simulation process, which starts with the simultaneous turning on of the 200 devices and adds 200 devices in each step, the average number of failed tasks in each step of up to 1600 devices in edge of IoT compared with this model in the cloud network is shown in fig. 6.

Increasing the number of end devices on the edge of the network, make the number of failed tasks more. As in the number of end devices more than 1500 of them are active devices, almost more than 60% of the tasks which sent to the edge for processing become failed duo to lack of resources, and increases the latency and reduces the efficiency of a large number of edge devices and utility gain of the system decreases. Therefore, as

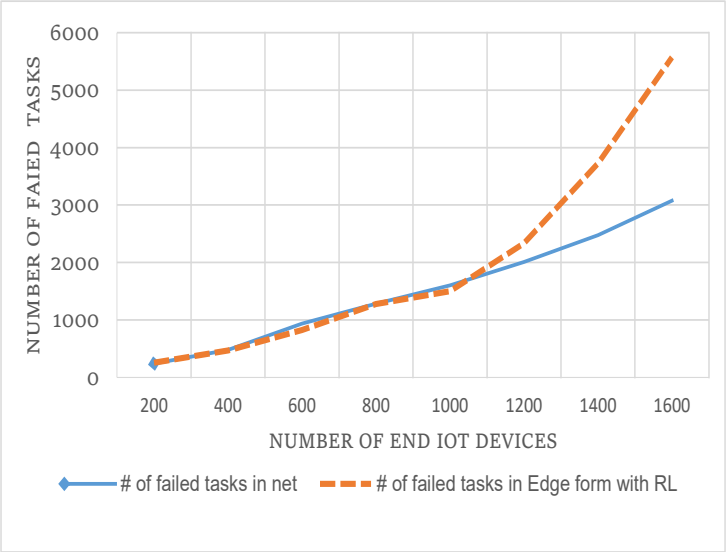

Figure 6: Number of failed tasks in cloud network vs. number of failed tasks in edge of IoT with RL method.

shown in the output of the simulation results in Fig. 9, this method works optimally in a few number of devices and prolongs the life of the final devices by making better use of energy sources. However, with increasing the number of devices while increasing the delay, due to the need to resend data and tasks, the battery energy of the end devices is consumed more.

According to the problem model in deciding to offload the task processing, especially in the number of end devices more than 1500, system faces to the overloading problem and negatively affect the offloading decision making and the network nearly goes to a standstill. But, by considering the limitations of the network edge terminals and device power supply limitations and limiting the number of terminals in terms of bandwidth required in real systems and adapting it to the simulation system, the overload problem is somewhat solved and the lifetime of end devices is increased.As shown in Fig. 7, by using the RL-based method in deciding to load the task processor into GW and the edge processing device, while the solution is aware, it also maximizes the benefit of the whole system and improves the decision in iterations again. The average gain of the whole system based on task processing at the edge in terms of the number of proportional devices (with balanced load), more than five times higher than cloud processing. In addition, the use of iterated base reinforcement learning method in network edge processing in repetition more than 1000 times is about 12.5% better than the normal edge processing method. Of course, it should be notice that the limit on the number of end devices at the edge of the network must be taken into account. Increasing in number of the active end devices, caused lack of the edge resources and deadlocked the task processing on edge of the network. In this simulation, the average energy consumption of the entire edge system is middling improved by 0.36 with the iterative reinforcement learning method compared to the processing transfer to the cloud

system[56, 58], and the lifetime of the final devices is increased.

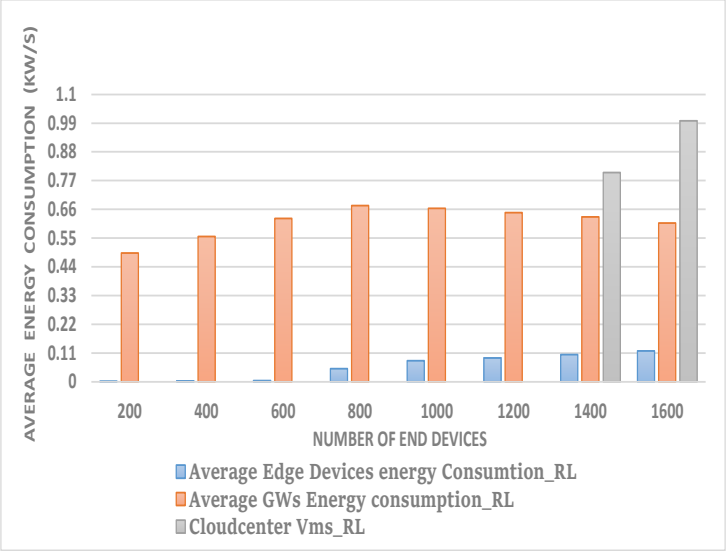

Figure 7: Average energy consumption in processing on edge of IoT with and with out the Q-RL method VS. cloud of this network with and with out the Q-RL method.

As the remote processing inputs and the volume of communication bandwidth between the end devices and the edge devices are constantly changing, as well as the rapid changes of the data of the end devices, so to avoid the standby energy consumption, which is comparable with the energy consumption of the working mode of the ballast device, in the simulation model, the on/off technique is used. This technique manages the system in a non-linear mode and brings simulation closer to the reality by applying idle-time and active-time in the operating model. In addition to this policy, along with the policy obtained from the RL method, it optimizes the energy consumption and increases the lifetime of the end devices of the IoT edge network at each stage.

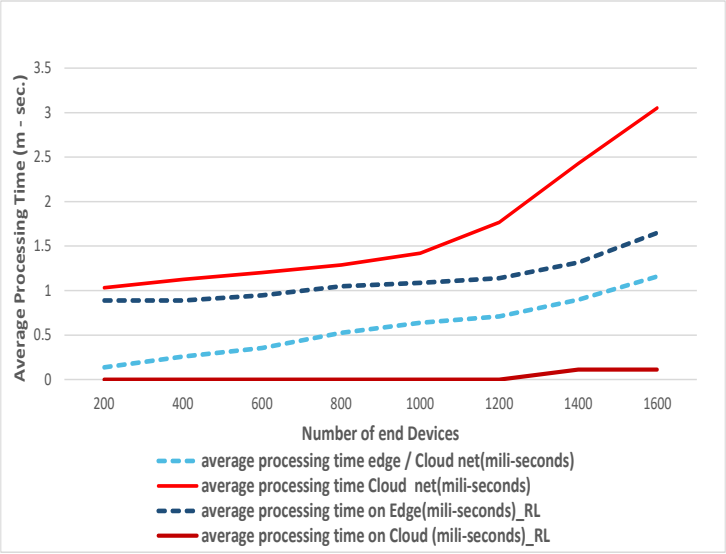

Figure 8: Average processing time on edge of IoT VS. cloud of this network with and with out the Q-RL method.

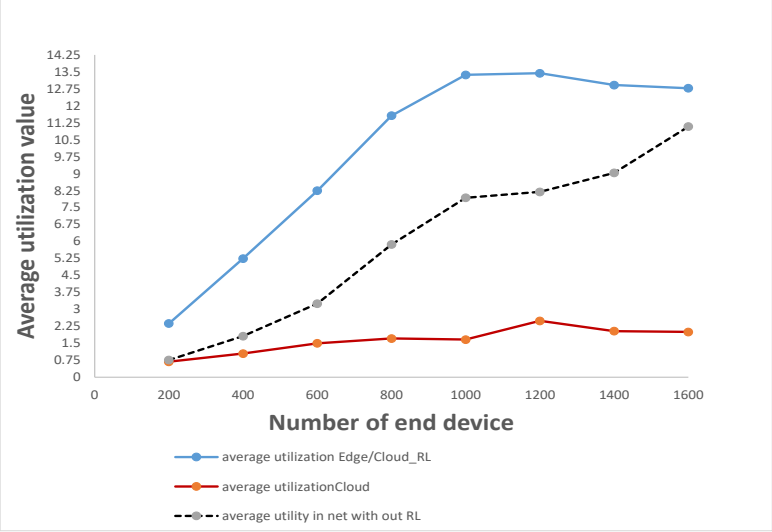

Figure 9: Average utilization value on edge/cloud System with and with out the Q-RL method.

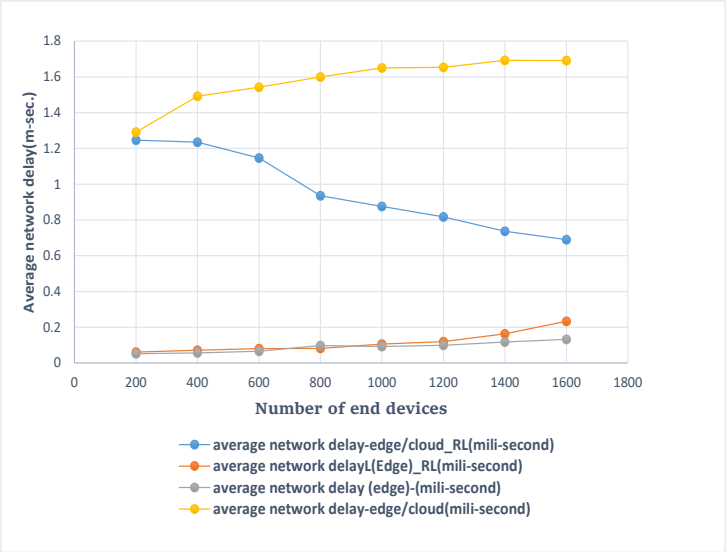

Figure 10: Average network delay on edge of IoT VS. cloud of this network with and with out the Q-RL method.

As shown in Fig. 8, Fig. 9 and Fig. 10; using the RL method and the consequential policy in this case, despite the increase in processing time at the edge of the network compared to the usual Edge/Cloud (Fig. 8) the total utility of the system is increased in Fig. 9. Also, the use of the RL method has significantly reduced the network latency. According to Fig. 10, using the edge model with RL method policy, result in less latency.
Another important point is evaluating the proposed algorithm based on reinforcement learning in task processing assignment. This model is employed to use the M task of the number of N users and to determine whether the best action over a given period of time is to out source processing or local processing. Given that the size of the input and output data of each ED device is assumed. The purpose is to extract a desirable process transfer function among the V policies. The overall size of the task assigned by $N * M$ is determined, and as the number of tasks increases, N increases in the Edge computing(EC)-IoT. The final number of 1500 users and five tasks for each user is assumed. Assuming the system parameters given for the simulation in Table1, the energy consumption in transmitted and received during the data offloading process for each edge system - which is grouped on average with 5 active devices and assuming 200 devices are turned on in each step - is $1.80 * 10^{-}6 J/bit$.

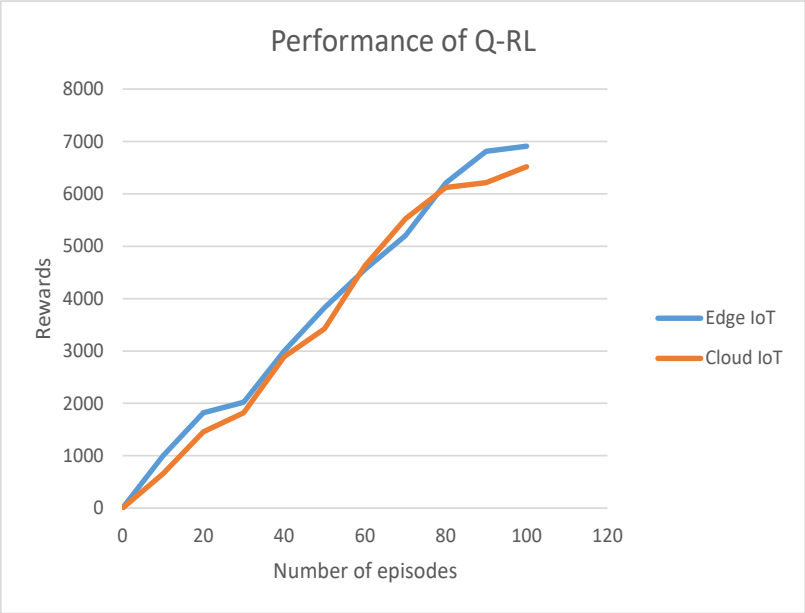

Figure 11: Performance of Q-Reinforcement Learning in edge of IoT VS. cloud of this network in 100 episodes.

Here the simulation model is taught to use to 100 episodes, the results of which showed that the use of off-policy Q-Learning-based reinforcement learning method improves edge systems performance and does not depend on the factor of explicit learning. As shown in Fig. 11, the RL-based system performs better in IoT edge processing than the same system in cloud processing. As the number of repetitions increases, the rate of improvement also increases. However, during the training process at the edge of the network, it was found that using RL is faster than the non-learning and classical mode. Central cloud devices, however, are more powerful. With more repetitions, the gap between edge processing and centralization increases, and the reward for the reinforcement learning method increases, while as the energy efficiency increases. The interest rate of the system under different parameters is shown in Fig. 9 in, which according to the learning rate and the weight rate of the parameters, it can be seen that the interest rate in the learning-based method is higher than the mode without using the above technique. Using the RL decision with Q-learning's off-policy in edge processing and local processing has better results in energy consumption, latency and gain. The main problem in modules designed with this model is in choosing the amount of learning and optimizing the parameters. Therefore, the proposed algorithm is tested under different learning rates, and after 100 iterations, It is found that the learning rate remains 0.005 and stable, and according to Fig. 11 it can be suitable for the mentioned target. Of course, for other values, the results were unstable and a lot of scatter was shown. The operating costs associated with the learning rates of 0.005 and 0.0005 were

examined and found that the ratio of the total cost to the learning rate of 0.005 is quite less than 0.0005. Of course, at the beginning of the training, there is a long distance due to the increase in operating costs. As the number of repetitions increases, this gap decreases. Of course, the costs in the last repetitions are almost equal. It is found that the use of Q-reinforcement learning works better and is more stable in several applications than the non-learner method and the central cloud method.

6. Conclusion

According to the increasing development of IoT end devices with limited size and high limitations in the required resources and increasing their number, the use of network edge resource management methods has become necessary. Also, according to the IoT network communication model and the access of edge devices to end devices and also higher level cloud devices, resource management solutions have been created at the edge of the network to reduce response time and efficient use of the resources. In this paper, it is assumed that there are several edge areas and network edge servers, and the system has multiple GWs, edge servers, and N to ED, with independent and instantaneous tasks in each ED. Any ED can be connected to EC via GW. Each task can be processed locally in ED or GW or remotely. Remote processing has three modes; Nearest edge server, adjacent edge server, remote cloud. A Q-RL algorithm is proposed to solve the optimization problem for deciding to assign tasks to one of the processing situations in order to reduce system costs such as energy consumption, transmission and processing latency, as well as the cost of loosing tasks. It is shown here that the Q-RL-based decision-making approach performs better in Edge-IOT than in Remote Cloud. Therefore, offloading to edge and adjacent edge servers has a high priority for offloading path selection. As shown in the simulation results and described in the problem model, the IoT network with the edge-cloud model encounters the NP-Hard computational problem in deciding to delegate processing. The main reason of this problem is the overload of the task and the limitation of network bandwidth. Of course, this problem can be solved to some extent by controlling the bandwidth capacity of the system and by using learning-based methods due to the limited life of the battery of end devices. But when the network gets too large, if the number of end devices becomes excessive instantly, the system almost crashes. Of course, to solve this problem, the on/off method is used to increase the policy of devices step by step using these methods and applying the optimal policy of the R-L method, to increases the utility of the system. The operating efficiency of the whole system is improved by 36% and the energy consumption at the edge of the network is optimized by 12.5%. It should be noted that the existing methods and even the method proposed in this article will encounter very serious problems in very large networks.As a future work different types of tasks with different processing lengths, classification steps, and task volume regression using Deep-RL and intelligent methods can be implemented. Another important issue is the effect of the dynamics and speed of mobility of devices at the end of the network edge which can increase the scope of investigations. However most of the challenges facing IoT systems in energy consumption and QoS requirements include workload variability, speed and accuracy of task assignment.

7. Conflict of interest and Authorship contributions

Conflict of interest

This manuscript is submitted for consideration of publication in Soft Computing journal. The manuscript is entitled "The Reinforcement Learning Method in Deciding on Task Offloading and Energy Consumption Management at the Edge of IoT Network". No fund was used in the research, and the authors have no affiliation with any organization with a direct or indirect financial interest in the subject matter discussed in the manuscript.
This paper has not been published elsewhere, also has not been submitted simultaneously for publication elsewhere. All authors of this manuscript have participated in conception and design, analysis and interpretation of the data.

Authorship contributions

This research includes the results of Ph.D thesis of Mr. Asghar Mohammadian, Ph.D candidate of Islamic Azad University, Science and Research Branch of Tehran,Tehran, Iran, and it has been done under supervision of Dr. Houman Zarabi and Dr. Amir Masoud Rahmani and the advice of Dr. Sam Jabbehdari at Islamic Azad University, Science and Research Branch of Tehran,Tehran, Iran. In this manuscript,A novel model of energy consumption management in IoT devices with the possibility of task offloading to edge devices of IoT network and improving energy consumption at the edge of IoT with using Q-Reinforcement Learning method by off-policy method in process offloading and energy consumption management is proposed. The IoT edge system is examined compared to the cloud network and the amount of the failed tasks and the delay caused by the transfer of tasks processing in the network is calculated. Then, the performance of the mentioned method at the edge of the network is evaluated relative to the same situation in the remote cloud from an operational point of view. It is shown that,the proposed method has a low latency in processing tasks at the edge of IoT and improves energy consumption.

References

[1] A. Ehsan, O. Anne-Cecile, L. Adrien, Estimating energy consumption of cloud, fog and edge computing infrastructures, IEEE Transactions on Sustainable Computing PP (2019) 1--1. doi:10.1109/TSUSC.2019.2905900.

[2] W. Li, T. Yang, F. C. Delicato, P. F. Pires, Z. Tari, S. U. Khan, A. Y. Zomaya, On enabling sustainable edge computing with renewable energy resources, IEEE Communications Magazine 56 (5) (2018) 94--101. doi:10.1109/MCOM.2018.1700888.

[3] W. Shi, J. Cao, Q. Zhang, Y. Li, L. Xu, Edge computing: Vision and challenges, IEEE Internet of Things Journal 3 (5) (2016) 637--646. doi:10.1109/JIOT.2016.2579198.

[4] X.-H. Lin, Y.-D. Huang, C. Yang, N. Xie, B. Chen, S. Zhang, H. Wang, An mdp based energy efficient transmission policy for wireless terminals, Communications and Network 05 (2013) 271--275. doi:10.4236/cn.2013.53B2050.

[5] A. H. Ismail, N. A. ElBahnasawy, H. F. A. Hamed, Agcm: Active queue management-based green cloud model for mobile edge computing, Wireless Personal Communications 105 (2019) 1--21. doi:10.1007/s11277-019-06119-1.

[6] P. Lai, Q. He, G. Cui, X. Xia, M. Abdelrazek, F. Chen, J. Hosking, J. Grundy, Y. Yang, Edge user allocation with dynamic quality of service, in: S. Yangui, I. Bouassida Rodriguez, K. Drira, Z. Tari (Eds.), Service-Oriented Computing, Springer International Publishing, Cham, 2019, pp. 86--101.

[7] T. Salem, S. Abd El-Kader, S. Abdel-Mageid, M. Zaki, Cognitive Networks:Applications and Deployments, 2014, pp. 3--42.

[8] Q. Wei, D. Liu, G. Shi, A novel dual iterative q-learning method for optimal battery management in smart residential environments, IEEE Transactions on Industrial Electronics 62 (4) (2015) 2509--2518. doi:10.1109/TIE.2014.2361485.

[9] B. B. Gupta, D. P. Agrawal, S. Yamaguchi, Deep learning models for human centered computing in fog and mobile edge networks, Journal of Ambient Intelligence and Humanized Computing 10 (2019) 2907--2911. doi:10.1007/s12652-018-0919-8.

[10] L. Xu, M. Qin, Q. Yang, K. Kwak, Deep reinforcement learning for dynamic access control with battery prediction for mobile-edge computing in green iot networks, in: 2019 11th International Conference on Wireless Communications and Signal Processing (WCSP), 2019, pp. 1--6. doi:10.1109/WCSP.2019.8927946.

[11] Y. Liu, C. Yang, L. Jiang, S. Xie, Y. Zhang, Intelligent edge computing for iot-based energy management in smart cities, IEEE Network 33 (2) (2019) 111--117. doi:10.1109/MNET.2019.1800254.

[12] M. Zeki-Suac, S. Mitrovi, A. Has, Machine learning based system for managing energy efficiency of public sector as an approach towards smart cities, International Journal of Information Management (2020) 102074doi:https://doi.org/10.1016/j.ijinfomgt.2020.102074.

[13] Y. Rioual, J. Laurent, E. Senn, J.-P. Diguet, Reinforcement Learning Strategies for Energy Management in Low Power IoT, in: CSCI, Las Vegas, United States, 2017.
URL https://hal.archives-ouvertes.fr/hal-01654931

[14] S. Kim, S. Park, Y. Kim, S. Kim, Vnf-eq: dynamic placement of virtual network functions for energy efficiency and qos guarantee in nfv, Cluster Computing 20 (3). doi:10.1007/s10586-017-1004-3.

[15] G. White, V. Nallur, S. Clarke, Quality of service approaches in iot: A systematic mapping, Journal of Systems and Software 132 (2017) 186 -- 203. doi:https://doi.org/10.1016/j.jss.2017.05.125.

[16] L. Li, M. Rong, G. Zhang, An internet of things qos estimate approach based on multi-dimension qos, in: 2014 9th International Conference on Computer Science Education, 2014, pp. 998--1002. doi:10.1109/ICCSE.2014.6926613.

[17] M. Yannuzzi, R. Milito, R. Serral-Graci, D. Montero, M. Nemirovsky, Key ingredients in an iot recipe: Fog computing, cloud computing, and more fog computing, in: 2014 IEEE 19th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), 2014, pp. 325--329.

[18] B. Rudra, D. S. Prashanth, Models and Algorithms for Energy Conservation in Internet of Things, Springer Singapore, Singapore, 2019, pp. 75--110. doi:10.1007/978-981-13-7399-2_4.
URL https://doi.org/10.1007/978-981-13-7399-2_4

[19] M. Jia, J. Cao, L. Yang, Heuristic offloading of concurrent tasks for computation-intensive applications in mobile cloud computing, in: 2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2014, pp. 352--357.

[20] C. Liu, M. Bennis, M. Debbah, H. V. Poor, Dynamic task offloading and resource allocation for ultra-reliable low-latency edge computing, IEEE Transactions on Communications 67 (6) (2019) 4132--4150. doi:10.1109/TCOMM.2019.2898573.

[21] K. Wang, Z. Xiong, L. Chen, P. Zhou, H. Shin, Joint time delay and energy optimization with intelligent overclocking in edge computing, Science China Information Sciences 63 (4). doi:10.1007/s11432-019-2780-0.

[22] C. Li, W. Chen, J. Tang, Y. Luo, Radio and computing resource allocation with energy harvesting devices in mobile edge computing environment, Computer Communications 145 (2019) 193 -- 202. doi:https://doi.org/10.1016/j.comcom.2019.06.001.
URL http://www.sciencedirect.com/science/article/pii/S0140366418307503

[23] F. Zhang, J. Ge, C. Wong, C. Li, X. Chen, S. Zhang, B. Luo, H. Zhang, V. Chang, Online learning offloading framework for heterogeneous mobile edge computing system, Journal of Parallel and Distributed Computing 128 (2019) 167 -- 183. doi:https://doi.org/10.1016/j.jpdc.2019.02.003.

[24] M. E. Khanouche, Y. Amirat, A. Chibani, M. Kerkar, A. Yachir, Energy-centered and qos-aware services selection for internet of things, IEEE Transactions on Automation Science and Engineering 13 (3) (2016) 1256--1269. doi:10.1109/TASE.2016.2539240.

[25] A. Carrega, G. Portomauro, M. Repetto, G. Robino, Balancing qos and power consumption in programmable 5g infrastructures, Transactions on Emerging Telecommunications Technologies 29 (11) (2018) e3425, e3425 ett.3425. arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/ett.3425, doi:10.1002/ett.3425.
URL https://onlinelibrary.wiley.com/doi/abs/10.1002/ett.3425

[26] Z. Hong, W. Chen, H. Huang, S. Guo, Z. Zheng, Multi-hop cooperative computation offloading for industrial iotedgecloud computing environments, IEEE

Transactions on Parallel and Distributed Systems 30 (12) (2019) 2759--2774. doi:10.1109/TPDS.2019.2926979.

[27] S. Li, J. Huang, Energy efficient resource management and task scheduling for iot services in edge computing paradigm, in: 2017 IEEE International Symposium on Parallel and Distributed Processing with Applications and 2017 IEEE International Conference on Ubiquitous Computing and Communications (ISPA/IUCC), 2017, pp. 846--851.

[28] V. G. Tharinda Nishantha Vidanagama, D. Arai, T. Ogishi, M2m gateway selection scheme for smart wireless devices: an energy consumption perspective, in: 2015 10th Asia-Pacific Symposium on Information and Telecommunication Technologies (APSITT), 2015, pp. 1--3. doi:10.1109/APSITT.2015.7217093.

[29] J. Mocnej, M. Mikuf, P. Papcun, I. Zolotov, Impact of edge computing paradigm on energy consumption in iot, IFAC-PapersOnLine 51 (6) (2018) 162 -- 167, 15th IFAC Conference on Programmable Devices and Embedded Systems PDeS 2018. doi:https://doi.org/10.1016/j.ifacol.2018.07.147.
URL http://www.sciencedirect.com/science/article/pii/S2405896318308917

[30] R. L. Gomes, L. F. Bittencourt, E. R. Madeira, Fuza: An algorithm for definition of reliable virtual networks to the edge as a service paradigm, J. Netw. Syst. Manage. 27 (2) (2019) 388408. doi:10.1007/s10922-018-9470-3.
URL https://doi.org/10.1007/s10922-018-9470-3

[31] F. Samie, L. Bauer, J. Henkel, From cloud down to things: An overview of machine learning in internet of things, IEEE Internet of Things Journal 6 (3) (2019) 4921--4934. doi:10.1109/JIOT.2019.2893866.

[32] V. Sharma, U. Mukherji, V. Joseph, S. Gupta, Optimal energy management policies for energy harvesting sensor nodes, IEEE Transactions on Wireless Communications 9 (4) (2010) 1326--1336.

[33] K. J. Prabuchandran, S. K. Meena, S. Bhatnagar, Q-learning based energy management policies for a single sensor node with finite buffer, IEEE Wireless Communications Letters 2 (1) (2013) 82--85.

[34] Y. Yin, L. Chen, Y. Xu, J. Wan, H. Zhang, Z. Mai, Qos prediction for service recommendation with deep feature learning in edge computing environment mobile networks and applications, Mobile Networks and Applications 25 (3) (2020) 391--401. doi:10.1007/s11036-019-01241-7.

[35] B. Martinez, M. Montn, I. Vilajosana, J. D. Prades, The power of models: Modeling power consumption for iot devices, IEEE Sensors Journal 15 (10) (2015) 5777--5789. doi:10.1109/JSEN.2015.2445094.

[36] G. Bedi, G. K. Venayagamoorthy, R. Singh, R. R. Brooks, K. Wang, Review of internet of things (iot) in electric power and energy systems, IEEE Internet of Things Journal 5 (2) (2018) 847--870. doi:10.1109/JIOT.2018.2802704.

[37] F. Kaup, P. Gottschling, D. Hausheer, Powerpi: Measuring and modeling the power consumption of the raspberry pi, in: 39th Annual IEEE Conference on Local Computer Networks, 2014, pp. 236--243. doi:10.1109/LCN.2014.6925777.

[38] A. Vishwanath, K. Hinton, R. W. A. Ayre, R. S. Tucker, Modeling energy consumption in high-capacity routers and switches, IEEE Journal on Selected Areas in Communications 32 (8) (2014) 1524--1532.

[39] J. Baliga, R. Ayre, K. Hinton, R. S. Tucker, Energy consumption in wired and wireless access networks, IEEE Communications Magazine 49 (6) (2011) 70--77.

[40] F. Jalali, K. Hinton, R. Ayre, T. Alpcan, R. S. Tucker, Fog computing may help to save energy in cloud computing, IEEE Journal on Selected Areas in Communications 34 (5) (2016) 1728--1739.

[41] K. Hinton, F. Jalali, A. Matin, Energy consumption modelling of optical networks, Photonic Network Communications 30 (16). doi:10.1007/s11107-015-0491-5.

[42] L. Pu, X. Chen, J. Xu, X. Fu, D2d fogging: An energy-efficient and incentive-aware task offloading framework via network-assisted d2d collaboration, IEEE Journal on Selected Areas in Communications 34 (12) (2016) 3887--3901.

[43] X. Chen, L. Jiao, W. Li, X. Fu, Efficient multi-user computation offloading for mobile-edge cloud computing, IEEE/ACM Transactions on Networking 24 (5) (2016) 2795--2808. doi:10.1109/TNET.2015.2487344.

[44] Y. Mao, J. Zhang, K. B. Letaief, Dynamic computation offloading for mobile-edge computing with energy harvesting devices, IEEE Journal on Selected Areas in Communications 34 (12) (2016) 3590--3605.

[45] X. Lin, Y. Wang, Q. Xie, M. Pedram, Task scheduling with dynamic voltage and frequency scaling for energy minimization in the mobile cloud computing environment, IEEE Transactions on Services Computing 8 (2) (2015) 175--186.

[46] X. Chen, Decentralized computation offloading game for mobile cloud computing, IEEE Transactions on Parallel and Distributed Systems 26 (4) (2015) 974--983.

[47] X. Lyu, H. Tian, W. Ni, Y. Zhang, P. Zhang, R. P. Liu, Energy-efficient admission of delay-sensitive tasks for mobile edge computing, IEEE Transactions on Communications 66 (6) (2018) 2603--2616. doi:10.1109/TCOMM.2018.2799937.

[48] D. Bertsekas, Multiagent value iteration algorithms in dynamic programming and reinforcement learning, Results in Control and Optimization 1 (2020) 100003. doi:https://doi.org/10.1016/j.rico.2020.100003.
URL https://www.sciencedirect.com/science/article/pii/S2666720720300035

[49] Y. Pochet, L. Wolsey, Production planning by mixed integer programmingdoi:10.1007/0-387-33477-7.

[50] F. Hussain, S. A. Hassan, R. Hussain, E. Hossain, Machine learning for resource management in cellular and iot networks: Potentials, current solutions, and open challenges, IEEE Communications Surveys Tutorials 22 (2) (2020) 1251--1275. doi:10.1109/COMST.2020.2964534.

[51] J. Lu, X. Guo, X.-g. Zhao, H. Zhou, A parallel tasks scheduling algorithm with markov decision process in edge computing, in: Z. Yu, C. Becker, G. Xing (Eds.), Green, Pervasive, and Cloud Computing, Springer International Publishing, Cham, 2020, pp. 362--375.

[52] S. Sheng, P. Chen, Z. Chen, L. Wu, Y. Yao, Deep reinforcement learning-based task scheduling in iot edge computing, Sensors 21 (5). doi:10.3390/s21051666.
URL https://www.mdpi.com/1424-8220/21/5/1666

[53] A. Defazio, T. Graepel, A comparison of learning algorithms on the arcade learning environment, CoRR abs/1410.8620. arXiv:1410.8620.
URL http://arxiv.org/abs/1410.8620

[54] D. Zhao, H. Wang, K. Shao, Y. Zhu, Deep reinforcement learning with experience replay based on sarsa, in: 2016 IEEE Symposium Series on Computational Intelligence (SSCI), 2016, pp. 1--6. doi:10.1109/SSCI.2016.7849837.

[55] G. A. Rummery, M. Niranjan, On-line q-learning using connectionist systems, Tech. rep. (1994).

[56] H. Gupta, A. Dastjerdi, S. Ghosh, R. Buyya, ifogsim: A toolkit for modeling and simulation of resource management techniques in internet of things, edge and fog computing environments, Software: Practice and Experience 47. doi:10.1002/spe.2509.

[57] R. Buyya, S. N. Srirama, Modeling and Simulation of Fog and Edge Computing Environments Using iFogSim Toolkit, Wiley, 2019, pp. 433--465. doi:10.1002/9781119525080.ch17.
URL https://ieeexplore.ieee.org/document/8654084

[58] Q. Zhiguo, W. Yilin, S. Le, P. Dandan, L. Zheng, Study qos optimization and energy saving techniques in cloud, fog, edge, and iot, Complexity 2020 (2020) 1--16. doi:10.1155/2020/8964165.

Biography

Asghar Mohamadian received his B.S. degree in computer engineering, and M.Sc. in mechatronic and computer systems from Islamic Azad University, Qazvin branch, Iran in 2006 and 2008. He also is a Ph.D. Candidate in computer system architecture in the Science and Research Branch, Islamic Azad University, Tehran, Iran from 2015, respectively. He is currently a Faculty Member of Islamic Azad University, Ilkhchi branch, Ilkhchi, Iran, and his research interests in Cloud and Edge computing, IoT, Deep learning, and autonomous vehicle systems.

Houman Zarrabi received his doctoral of engineering from Concordia University in Mon-treal, Canada in 2011. Since then he has been involved in various industrial and research projects. His main expertise includes IoT, M2M, big data, embedded systems and VLSI. He is currently the national IoT program director and a faculty member of ITRC.

Sam Jabbehdari currently working as an associated professor at the department of Computer Engineering in IAU (Islamic Azad University), North Tehran Branch, in Tehran, since 1993. He received his both B.Sc. and M.S. degrees in Electrical Engineering Telecommunication from Khajeh Nasir Toosi University of Technology, and IAU, South Tehran branch in Tehran, Iran, respectively. He was honored Ph.D. degree in Computer Engineering from IAU, Science and Research Branch, Tehran, Iran in 2005. His current research interests are Scheduling, QoS, MANETs, Wireless Sensor Networks and Cloud Computing.

Amir Masoud Rahmani received his BS in Computer Engineering from Amir Kabir University, Tehran, in 1996, the MS in Computer Engineering from Sharif University of Technology, Tehran, in 1998 and the PhD degree in Computer Engineering from IAU University, Tehran, in 2005 .Currently, he is a Professor in the Department of Computer Engineering at the IAU University. He is the author/co-author of more than 200 publications in technical journals and conferences. His research interests are in he areas of distributed systems, ad-hoc , Internet of Things , and wireless sensor networks and evolutionary computing.