Computers and Electrical Engineering 72 (2018) 348–360

Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng

Task offloading and resource allocation for edge-of-things computing on smart healthcare systems☆

Kai Lin*, Sameer Pankaj, Di Wang

Dalian University of Technology, No.2 Linggong Road, Ganjingzi District, Dalian City, Dalian, Liaoning 116024, China

ARTICLE INFO

Article history:
Received 30 December 2017
Revised 4 July 2018
Accepted 1 October 2018
Available online 9 October 2018

Keywords:
Edge-of-things computing
Healthcare system
Task offloading
Resource allocation
Energy efficiency.

ABSTRACT

Impelled by prevalent smart devices and omnipresent wireless communication networks, Edge-of-things transpires as a captivating paradigm to accommodate power-sensitive or compute-intensive applications over resource-constrained smart devices. In this research, we focus on flexible compute-intensive task offloading to a local cloud (i.e., cloudlet) saving energy, which aims to optimize the energy consumption, the operation speed, and the cost. A fruit fly optimization based task offloading algorithm (FOTO) is proposed, which improves offloading and resources allocation to acquire the nominal energy consumption under the existing restraints. Performances are evaluated regarding energy consumption, execution time and cost, which are compared with the cooperative multi-tasks scheduling based on ant colony optimization algorithm (CMS-ACO) and heuristic queue based algorithm (GA-ACO). The experimental results prove the effectiveness of proposed FOTO algorithm by comparing with existing algorithms.

1. Introduction

The eruptive growth of mobile phones and smart devices accompanying with the swift development of wireless networks and mobile computing technologies have moved the emergence of numerous novel computing models, such as fog computing, mobile cloud computing (MCC), edge computing etc. [1]. Edge computing refers to the empowering technologies allowing computations to be performed at the edge of the network, upon down streaming and up streaming data on behalf of cloud services and internet-of-things (IoT) services, respectively. For example, a smart device is an edge between body things and cloud; a gateway in a smart home is an edge between home things and cloud; finally a cloudlet [2] is an edge between a smart device and cloud. By using the cloud as a centralized server just rises the frequency of communique between user devices, (e.g. smart devices, tablets, wearable devices, and gadgets) we denote these as edge devices, and physically faraway cloud data centers. This limits the applications requiring real time acknowledgement. Mobile edge computing in a fanatic-dense network is anticipated to be an efficacious explanation which meets the little latency demands [3,4].

Healthcare organizations are strongly adopting analytic solutions as a part of their health information technology (HIT) infrastructures to provide better healthcare services. To identify invalid signatures more efficiently, a batch recognition game system in wireless mobile networks was developed allowing nodes to discover invalid signatures with rational delay irre-

☆ Reviews processed and recommended for publication to the Editor-in-Chief by Guest Editor Dr. M. M. Hassan.

* Corresponding author.
E-mail address: link@dlut.edu.cn (K. Lin).

https://doi.org/10.1016/j.compeleceng.2018.10.003
0045-7906/

spective of if the game scenario contains comprehensive information or inadequate information [5]. Edge computing has been becoming more famous in healthcare since organizations introduced the number of linked medical devices into their HIT ecosystem. The aim of edge computing is to reduce the latency by bringing the public cloud abilities to the edge [6,7]. Offloading is an explanation to enhance these mobile schemes competences by relocating computation to added ingenious computers (i.e., servers). This is disparate from the traditional client-server infrastructure, where a thin client continuously migrates computations to a server [8,9]. In mobile devices, the goal of computation offloading provides the cost minimization. In this study, the computation offloading as the service and the key of computation offloading as a service for mobile devices (COSMOS) enables the intermediate service between the mobile devices and the corresponding cloud service provider [10]. The communication resources are transparent to the mobile devices to recognize the resources cost-effectively, hence satisfying the demand for mobile devices. A mobility aware content positioning infrastructure was presented by Chen et al. [11], where precise caching policies were established for SBSs in 5G and smart device leverage user movement, aiming to enlarge the cache hit ratio.

The proposed offloading model addresses the problem of resource constraint. The proper allocation of cloud resources to the smart devices is important because of the low resource capacity in smart devices. The proper allocation provides better throughput of the applications. Therefore, a new multi-objective mechanism is proposed for resource scheduling. At first, a regression algorithm is presented to reformulate the user requests in order to avoid the repeated request of the same user. Secondly, a mathematical model is defined to solve the resource allocation problem. Finally, the performances are evaluated and compared with existing offloading algorithms. Three objective functions are used in this proposed algorithm to enhance the novelty of the work in task offloading. The main contributions of this article are as follows:

- Proposing edge-of-things computing based task offloading and resource allocation for smart devices in healthcare systems.
- Providing the shortest task completing time, lowest energy consumption and minimum cost while comparing with the existing task offloading approaches in the healthcare systems.
- Giving a pragmatic approach to take benefit of edge-of-things computing in smart healthcare systems.
- Depicting the characteristics of edge-of-things computing based healthcare systems and its assistance from different perspectives.
- Demonstrating system model with three layered architecture in healthcare system
- Providing an efficient methodology for the task offloading approach in the smart devices and healthcare system as well.

The remainder of this paper is organized as follows: A brief discussion about related work is made in Section 1, which gives the explanation of task offloading using various approaches. Later, Section 3 presents problem formulation and system model to study the schematic representation of proposed work. Then, Section 4 gives the proposed methodology in which the mathematical model is designed and the objective functions are defined to solve the task offloading problem. Finally, Section 5 gives the evaluation of results and analysis of proposed work with other existing methods. Lastly, Section 6 brings the conclusion.

2. Related work

The cloud resource allocation was described by Chunlin et al. [12]. The energy consumption is optimized by the mobile cloud resource allocation approach. The resource allocation was performed on the public and local cloud level. The performance was evaluated in the experimental environment, for that the algorithm of resource allocation was proposed and the compared results were analyzed. In mobile devices, the battery is considered as the main component, and it was limited to modern mobile devices. It mainly occurred in video application and was pointed out by Zhang et al. [13]. This was very complicated with bandwidth and delay constraints. In state of the art and wireless network channel platform, the energy efficiency and the performance were examined. The opportunities and the challenges were identified for real-time video application. In dynamic wireless network conditions, the scheduling algorithm made the offloading decision over the trace-driven simulations.

In healthcare industries, to deliver a more suitable service and environment, a cyber-physical scheme for patient centric healthcare applications and facilities, known as Health-CPS was constructed on cloud and big data analytics skills by Zhang et al. [14]. This system was contained of a data gathering phase with a united standard, a data organization phase for dispersed storage and parallel computing along with a data-oriented service phase which shows that the skills of cloud and big data can be cast-off to boost the performance of the healthcare system, so that humans can relish numerous smart healthcare applications and services. A smart personal health advisor (SPHA) for comprehensive and smart health monitoring and supervising system was developed by Chen et al. [15]. The SPHAS core infrastructure was proposed to evaluate the overall health status of the user. The energy consumption issues had caused a stir in cloud computing. In this consolidation, multi-core processors and virtualization were important. This study has enhanced the model of green cloud scheduling to exploit the resource and heterogeneity of tasks to allocate the scheduled tasks.

None of the previous works had devised a proper solution to energy constraint in healthcare devices. Hence, we proposed a novel cloudlet embedded edge-of-things paradigm having better task offloading algorithm which focuses on three key parameters; energy consumption, execution time and data center cost. The proposed task offloading algorithm has shown that it is beneficial and enhances the smart devices abilities in healthcare systems.

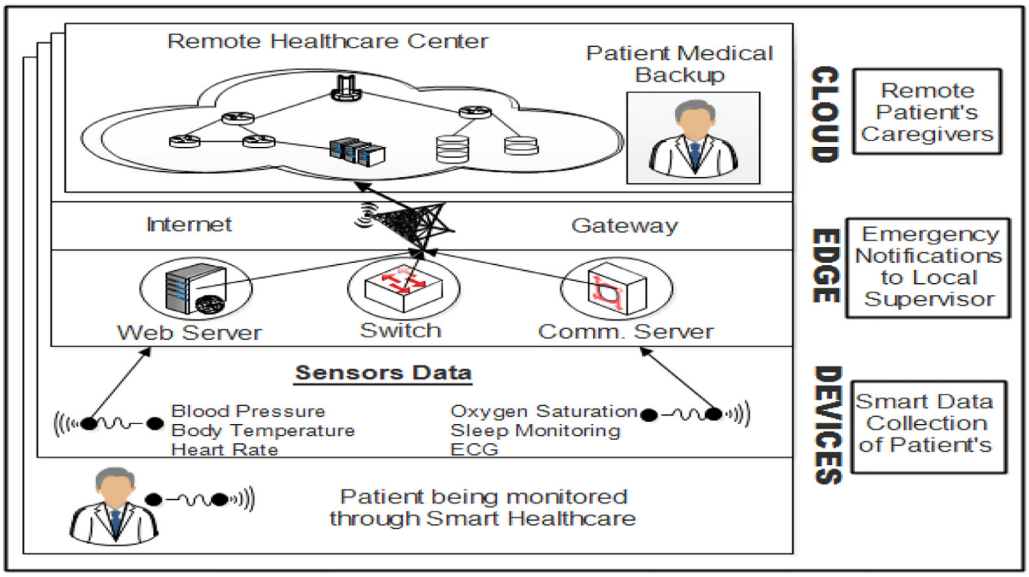


Fig. 1. Three-layered architecture of edge computing in Healthcare.

3. System model and problem statement

The edge computing in our proposed work consists of three layers: devices, edges, and clouds. Cloudlets are adaptability-intensified micro-data centers situated at the edge of a network and usually perform the smart devices part of the network. These are designed to handle resource-intensive smart device applications and to take the load off from both the network and the central data center and to keep computing near to the point of origin. The cloud here is a great place for centralized computing, but not every computing task is needed to run on a centralized system. Edge computing is demonstrated by one or more layers of edge nodes which are placed between the cloud and the edge devices and surrounded by gateways near to the field. From Fig. 1. The device users' service requests are considered as a task in the cloudlet and this task is assigned to a node in a cloudlet. The users offload their requests to the nearest cloudlet in their geographical zone (i.e. host cloudlet) because the cloudlets are spread over various geographical areas. For processing every request from the users, the cloudlet proxy read the network status and chooses the suitable place, optimized with QoS parameters.

3.1. Graph model of task

The properties of directed acyclic graph (DAG) is followed in the smart device environment for task submission. The DAG is shown as $G = \{V,E\}$, where, V is the submitted job which contains tasks $T = \{T_1, T_2, \ldots T_N\}$ and E is the set of connection between any two tasks T_i and T_j. Every node $i \in V$ in graph G illustrates a task and a directed edge $E(i,j)$ demonstrates the priority imperative between tasks i and j which means that task j cannot begin execution until its precedent task i is finished.

3.2. Problem formulation

Let's assume that there are 'j' computation tasks generated by the smart device users in the FOTO system at time period t. The maximal response time with the optimal resource of FOTO against users task 'j' can be gained by resolving the subsequent optimization problem. The fundamental thought of this work is accomplishing a decent offloading presentation at little financial cost by distribution of cloud resources among smart devices. This effort aims to minimize the cost of energy on cloud resources. With the improvement of distributed computing and the consistently developing number of smart devices, numerous applications need higher client's quality of experience (QoE). The objective functions are used to minimize the overall task completion time, energy consumption and cost.

Problem:

$$min f(x) = \sum_{i=1}^{k}\sum_{j=1}^{m} T_i V M_k \tag{1}$$

$$s.t. \sum_{i=1}^{k} dl_i, i = 1, 2, \ldots, k \tag{2}$$

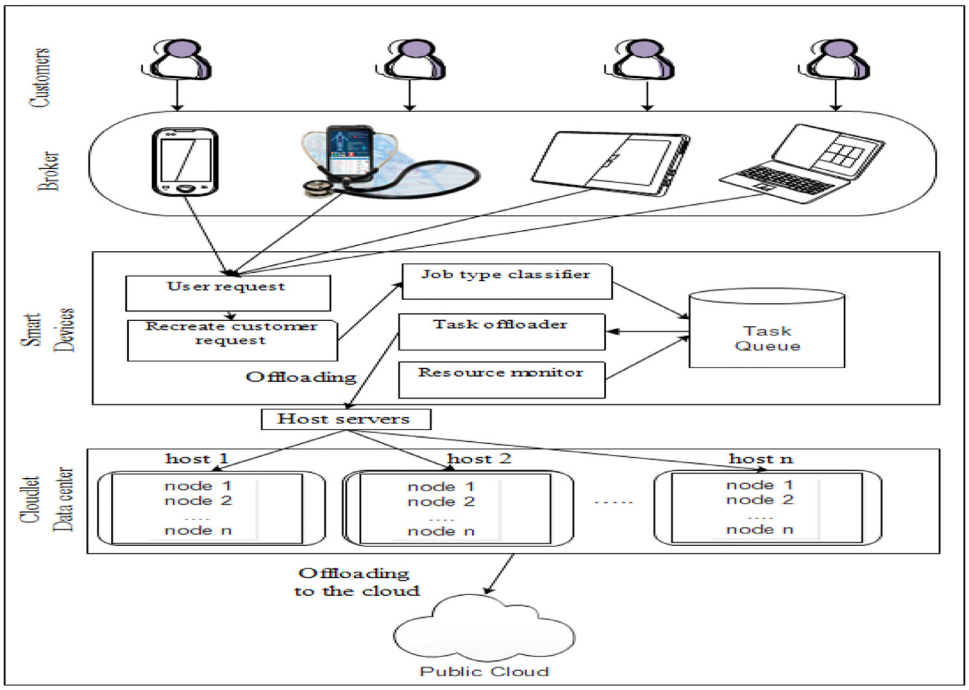


Fig. 2. : Proposed system model.

and

$VM_K = \mathrm{D}$,

$$Where, D : \wedge_{k \in K}\left(\sum X(k, D) = 1\right) \tag{3}$$

In the Eq. (1), $f(x)$ is the multi-objective functions of the optimization problem in proposed model. The functions $f_1(x)$, $f_2(x)$ and $f_3(x)$ are used to find the overall task completion time, the energy consumption and the cost, respectively, which finds the optimum results. Whereas i and j are the number of virtual machines upto k and m, respectively, which are used in the objective functions. In the first constraint, dl_i is the deadline for T_i task execution. The second constraint represents that each VMk is mapped to the data center D that is mapped only once. In this statement, $X(k, D)$ is the integer variable equal to 1, if VM is assigned to the data center, otherwise it is 0. And k belongs to specific VM while K belongs to all VM in the Virtual Machine list. The above-mentioned statements are explained under task offloading section. The task offloading process is used here to mitigate the energy consumption and workload of a smart device while the tasks are moved to the execution of cloud.

4. FOTO methodology for task offloading

A short depiction of proposed FOTO strategy is given in this section. A flexible compute-intensive task offloading plan from smart devices to cloudlet for energy savings is proposed. This comprises of three segments. (i) Device manager, which is in charge of keeping up the nodes capabilities in a cloudlet. (ii) Broker, which is responsible for allocating tasks to nodes in a cloudlet based on the nodes capability. (iii) Task tracker, which checks the execution of the task in nodes. Linear regression algorithm analyses the requests received from the user by the broker. The jobs are clustered with pipeline tree classifier approach and placed in a corresponding queue. Fruit Fly optimization algorithm obtains the optimum resource for the task. The unwanted nodes are placed in idle mode after the completion of the task. Thus, the energy consumption will be reduced. The detailed scheme of steps involved in proposed method are illustrated in Fig. 2.

4.1. Linear regression algorithm

Here, the customer requests are reformulated with this algorithm with the prediction of wasted resources based on the request. This regression analysis predicts the random variable value p and this value is based on the independent variable value q by least squares linear regression equation. A simple linear regression is represented as [16]:

23

$$p = d_0 + d_1 q \tag{4}$$

Where p is the predictable variable, q is the input variable. d_0, d_1 are the coefficients of linear regression algorithm which we needed to estimate. The aim is to attain the finest approximations for the coefficients to minimalize the mistakes in predicting p from q.

Aforementioned statement determines that how these conditions used to evaluate the resources quantity and on the basis of request history created by the same client. The Eq. (5) is used to decide the estimation of the CPU used by the client as a function of the present and additionally, the past requests made by a similar client [16]:

$$E_R = Sgn(C_R - W_R) + k \tag{5}$$

In the Eq. (5), E_R is the estimated request of the user, C_R is the current request of the user and W_R is the wasted resource and function 'Sgn' covers both the negative and positive parts of requests. While, k is the regression parameter whose value decides the projected values of the resources to be cast-off [16]. The cloud providers are provided control over the trade-off between reserving resources and incurring service level agreement (SLA) violations when the value of the k is set up properly. The wasted resources are computed as follows:

$$W_R = C_R.b_1 + b_0 \tag{6}$$

Where b_1 and b_0 are the coefficients of regression algorithm which are determined by using [16] and conditions (7) and (8):

$$b_1 = \frac{\sum\left[\left(C_{Ri} - \bar{C_R}\right).\left(W_{Ri} - \overline{W}_R\right)\right]}{\sum\left[\left(C_{Ri} - \bar{C}_{Ri}\right)^2\right]} \tag{7}$$

$$b_0 = \overline{W}_R - b_1.\bar{C}_R \tag{8}$$

In above condition, $(\bar{C}_{Ri})$ notifies the mean value of currently requested and $(\overline{W}_{Ri})$ notifies the mean value of wasted resources. i represents the number of current requests and number of wasted resources.

4.2. Job type classification

It helps to divide the incoming requests into various types and to classify the jobs of user queries. The incoming request contains several tuples and attributes. The job type is classified using pipelined decision tree classifier algorithm based on the tuple and attribute [17]. For this propose, pipelined decision tree classifier is used utilizing Google data flow in two stages: training and classification. A decision tree is constructed during the training stage with the use of modified iterative dichotomiser-3 (ID3) algorithm [18]. This ID3 is a decision tree induction approach for classification. Then, Fouriers transform is used to execute the data set (D) consisting of a number of attributes (A). The best attribute is the one giving maximum information (Info) gain. It is a mathematical way to predict the amount of information needed by picking a particular attribute. Info is the information gain and InfoA is the information gain of the attribute A. The pipeline path is achieved by running the pipeline-object.run () method beginning the pipeline service [17]. The pipeline classifier is executed using the mentioned function. For work execution, we use dataset for user request which contains class labels and set of attributes. These attributes are explained below and are classified using this pipeline algorithm.

Generally, a job request consist of a tuple like < Num_VM,RAM, Storage, Bandwidth, Execution_time, S_Time, E_Time >, where Num_VM is the number of VMs required, RAM is memory in megabytes, Storage is disk space in megabytes, Bandwidth is the network bandwidth in megabytes per second, Execution_Time is the execution time, S_Time is start time, and F_time is finish time. Since these are distinct requirements of each job, the request tuple can be used to identify the type of a submitted job as follows. A request is described to identify the type of the job by pipeline tree classifier,

Type 1 Job request = < Num_VM, Ram, Storage, Bandwidth, Execution_Time, S_Time, F_Time >
Type 2 Job request = < Num_VM, Ram, Storage, Bandwidth, Execution_Time, S_Time, Nil >
Type 3 Job request = < Num_VM, Ram, Storage, Bandwidth, Execution_Time, Nil, Nil >

4.3. Mathematical model

Task offloading problem is dealt with m numbers of hosts. The hosts in a data center are represented by the set of *Host = {h1,h2,...,hm}* with n numbers of nodes *Node = {n1,n2,...,nn}* and j tasks *T = {T1,T2,...,Tj}*. The user submits the tasks to the broker. The task is denoted by a group of tuples like $T_i = \{T_a, T_m, T_f\}$ where T_a is the arrival time, T_m is the memory and T_f is the finish time. The submitted task is plotted to the VM via a broker. In this method, we primarily focused on the usage of the virtual machines, the completion time of the tasks, energy consumption, and the data center cost.

Fruit fly optimization based task offloading (FOTO): The FOTO algorithm is designed based on fruit fly optimization algorithm (FOA) [19] which uses the tasks as fly to offload them to the cloudlet which searches for the suitable virtual machine based on the multi-objective function to attain the better execution time, less energy consumption and lower cost. The basic constraints are followed by the load of the virtual machine, after assigning the task, it should not be greater than the upper threshold value while selecting a suitable virtual machine for the removed task. FOA is known as the new meta-heuristic intelligent optimization algorithm which has been widely applied in various fields. The details of FOTO algorithm are depicted in Algorithm 1.

Algorithm 1 Pseudo of task offloading.

Initialization FF_S_j is the jth swarm of Fruit Fly
Attain $VM_{list}(ul) \leftarrow$ Null && $VM_{list}(ol) \leftarrow$ Null
For every VM in the data center do
Calculate the capability & load
    If $L_k > max\_capacity$ then
        There is no feasibility of load balancing and rearrange the load
    Else If $L_k > Th_{lower}$&&$L_k < Th_{upper}$ then
        Stable load for entire virtual machines
    Exit
    End if
For every j in number of Fruit Fly swarms (Smell-stationed search)
    Produces S fruit flies FF_{jk} (k=1,2,...,S) on jth Fruit Fly swarm
    For every VM_k in VM do
        If $L_k < Th_{upper}$ then
            Attain $VM_{list}(ol) \leftarrow VM_k$
            Else attain $VM_{list}(ul) \leftarrow VM_k$
        End if
    End for
End for
For every j in number of FF swarms do (vision stationed search)
    Assess the fruit flies produced FF_{jk}
End for
For every VM_k in VM do
    If $L_k ! = \phi$ then
        Categorize entire VM in increasing order;
        Categorize entire task stationed on the deadline;
For every T_j in $VM_{list}(ol)$ do
    Discover finest VM in $VM_{list}(ul)$ such that $L_k + ln_i \geqq Th_{lower}$ && $Lk + ln_i < Th_{upper}$ [finest position of Fruit Fly swarm (sub-population)-vision]
End for
Move T, best VM;
Else
Move VM_k to sleep mode;
End if;
Update T,L_k, VM_{list};
    End for
End for

Table 1
Simulation parameters.

Number of hosts	10
Number of VM	50
VM types	Small/Medium/Large
MIPS	1000/5000/10000
RAM	1820/3830/7530
Bandwidth	1000/1000/1000
Number of queries	100 to 1000

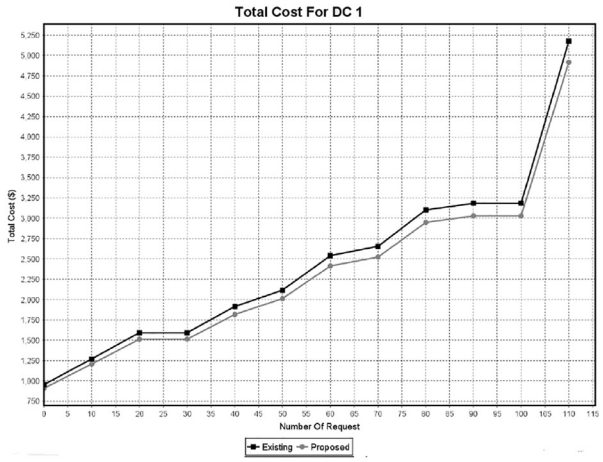
Now the processing time for all tasks P_j can be computed as [1]:

$$P_j = \sum_{i=1}^{j} P_{ik} k = 1, 2, \ldots, n \tag{9}$$

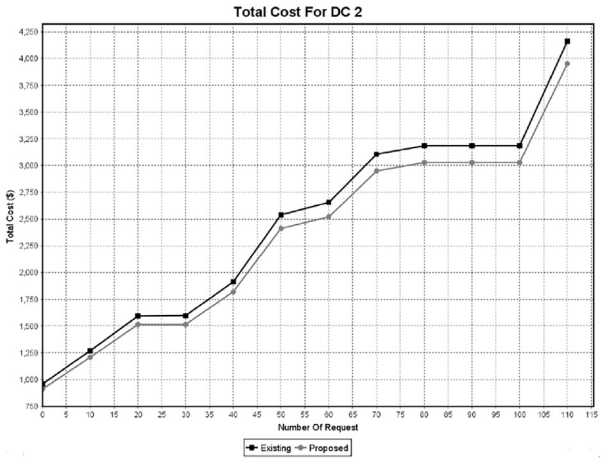
In the condition (9), P_{ik} is the processing time of task T_i on virtual machine VM_k and the value of the $T = 1, 2, \ldots, j$. The capacity of a virtual machine is calculated as [20]:

$$C_k = P_e(num - ck).P_e(mips - ck).VM_{bw} \tag{10}$$

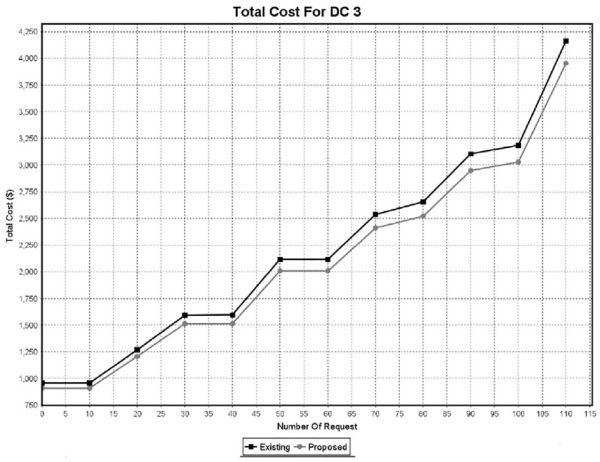
In the condition (10), capacity is defined as C_k (VM capacity of host with $k = 1, 2, \ldots, n$). The processing element is denoted by P_e, $Pe(num - ck)$ represents the count of the processor, $P_e(mips - ck)$ represents the processor (mips) and VM_{bw}



(a)Cost of data center 1.



(b)Cost of data center 2.



(c)Cost of data center 3.

Fig. 3. Cost of data center.

represents the bandwidth of VM [21]. Total length of tasks that are allocated to a VM is computed by:

$$L_k = \frac{N(T,t)}{S_r(k,t)} \tag{11}$$

In the condition (11), L_k is a load of a specific virtual machine, $N(T,t)$ is the total number of tasks at time t, $S_r(k, t)$ is the service rate of the virtual machine at time t. And j is the count of task while k is the virtual machine count.

Processing time of VM:

$$T_p(k) = \frac{L_k}{C_k} \tag{12}$$

In the condition (12), $T_p(k)$ is the processing time of virtual machine, $k = 1, 2, \ldots n$ and C_k is the capacity of virtual machine $k = 1, 2, \ldots n$.

Execution time of task *T*:

$$T_e = \frac{L_T}{C(VM_k)} \tag{13}$$

In the above condition (13), T_e is the required time for total execution, L_T represents the length of task T and the performance of CPU fraction is measured by $C(VM_k)$ [20].

The time to finish the task T_i on virtual machine VM_k is calculated as:

$$T_f = T_s + T_e \tag{14}$$

Where Ts is the starting time of task T_i on the virtual machine VM_k and T_e is the required time for total execution.

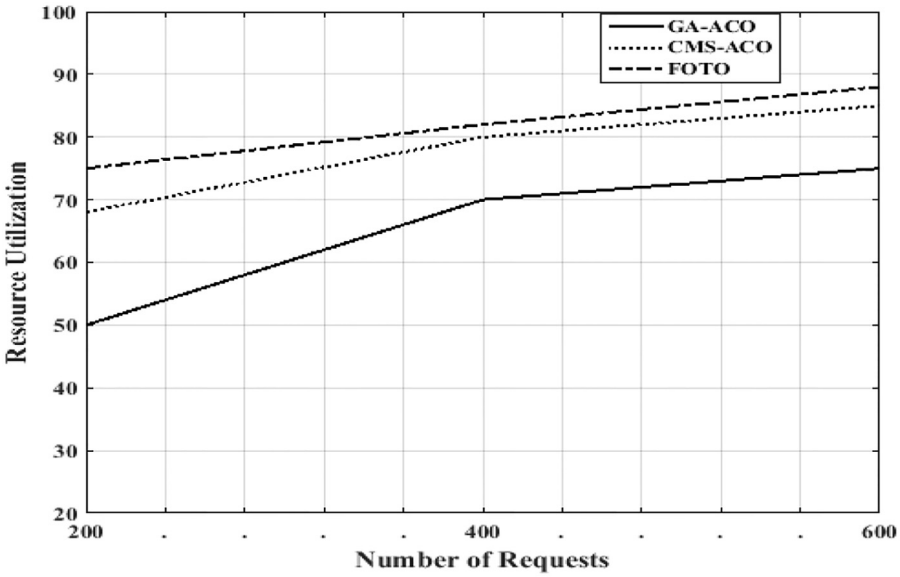


Fig. 4. Resource Utilization.

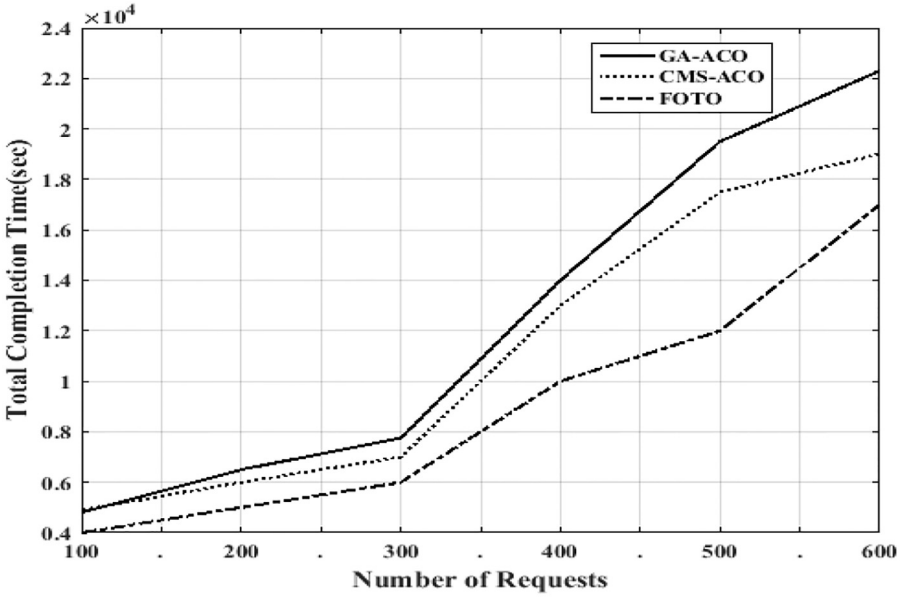


Fig. 5. Task Completion Time.

In below condition (15), δ_{ik} is a decision variable implemented because every task should be allocated to only one virtual machine [1]:

$$\delta_{ik} = \begin{cases} 1, & \text{if is assigned to the } VM_k \text{ and if } T_f < dl_i \\ 0, & \text{otherwise, if } T_f > dl_i \end{cases} \tag{15}$$

Where dl_i is the deadline to execute the task T_i. Therefore, the above condition defines the decision variable where the finishing time is less than deadline, which means the task will be allocated otherwise not. Thus, the objective functions are defined as:

Overall Task Completion Time: This objective function is implemented to diminish the overall completion time of task which is represented as [22]:

$$f_1(x) = min\left\{max_{T_j \in T, VM_k \in VM}, T_f\right\} \tag{16}$$

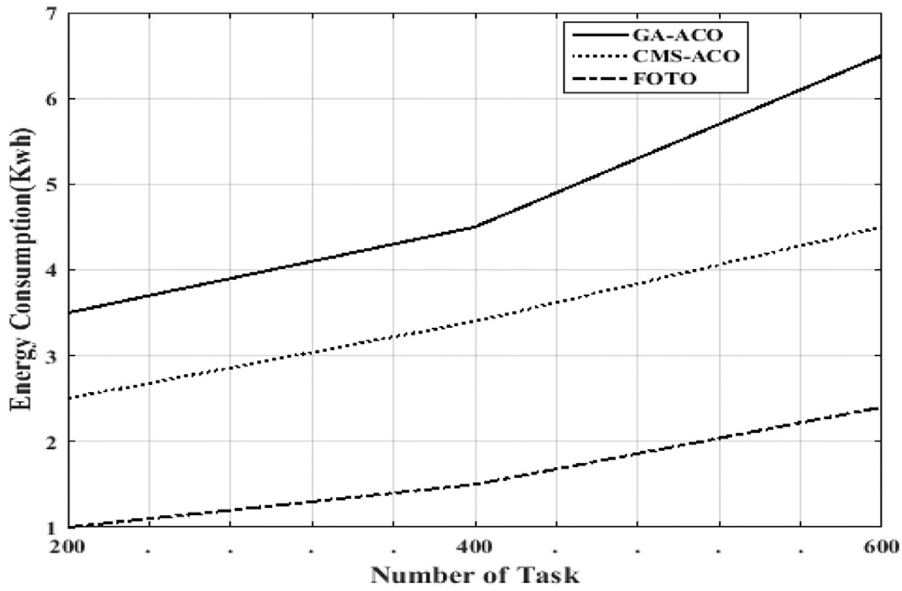


Fig. 6. Energy Consumption.

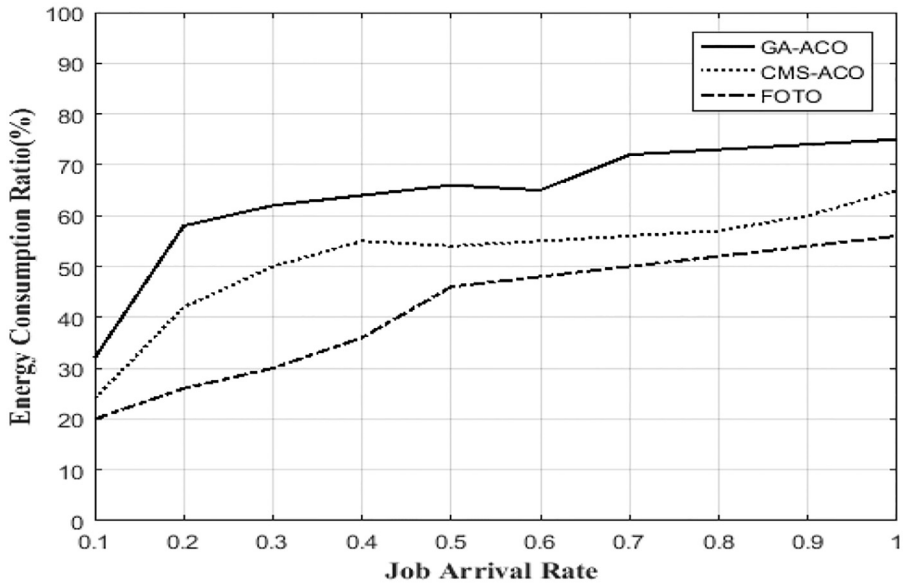


Fig. 7. Energy Consumption Ratio.

Where T_f is finishing time of task (T_i) on the virtual machine VM_k. And j is the count of the task. $T_j \in T$ represents all tasks and represents all VM's.

Energy Consumption: Consider EC_{jk} is the energy consumed by the task running on the *VM*. EC_r denotes the rate of energy consumption of the VM and T_e is the execution time of the task. The energy consumption (EC) is calculated as [22]:

$$EC_{jk} = EC_r.T_e \tag{17}$$

The total **Energy Consumption** is computed as follows:

$$f_2(x) = \sum_{i=1}^{j}\sum_{n=1}^{k} EC_{jk} \tag{18}$$

In the condition (18), there are j numbers of tasks used and k numbers of VM used. Whereas Σ represents the sum of energies consumed by a number of tasks and of virtual machines in offloading tasks.

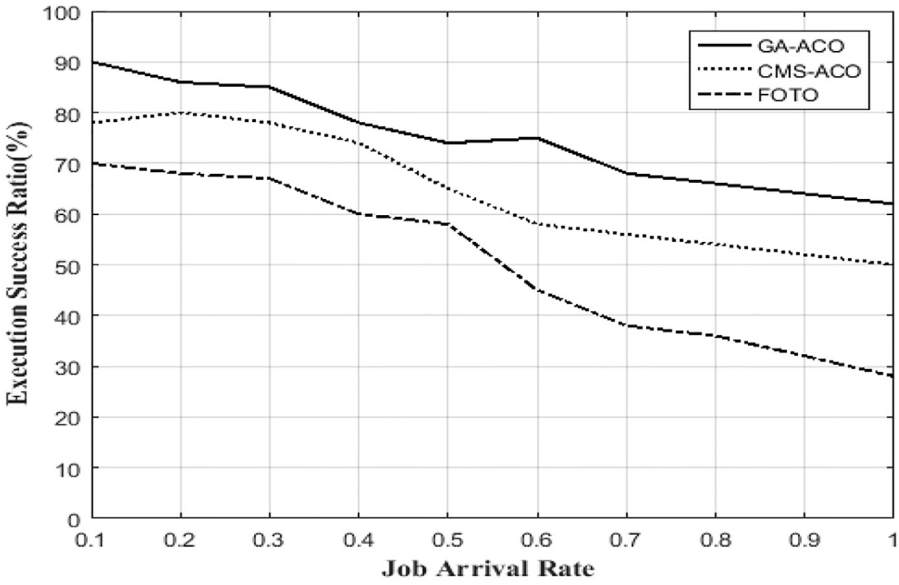


Fig. 8. Execution Time Ratio.

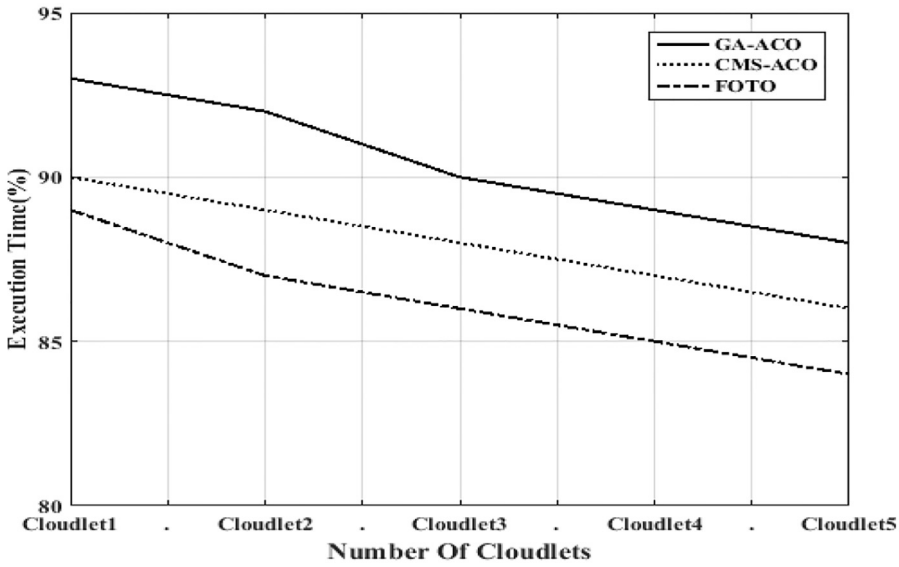


Fig. 9. Execution Time of Cloudlets.

Where $f_2(x)$ is the total energy consumption of the VM.
Cost: The cost of the data center is obtained as [12]:

$$f_3(x) = c.f_2(x) \tag{19}$$

Where c denotes the cost of 1 kW power and $f_3(x)$ is the cost of the data center.

5. Simulation setup

A novel FOTO algorithm is proposed for energy efficient task offloading in smart healthcare systems based on FOA with the help of Edge-of-things computing and cloud computing. We executed the algorithm by using CloudSim 2.0, an extendable toolkit for modelling and simulation of cloud computing environments that helps in modelling of VMs on an imitated node of a data center. The effectiveness and efficiency of the proposed approach are assessed primarily regarding the mean service response time and the mean energy consumption by smart devices. The implementation has been accomplished

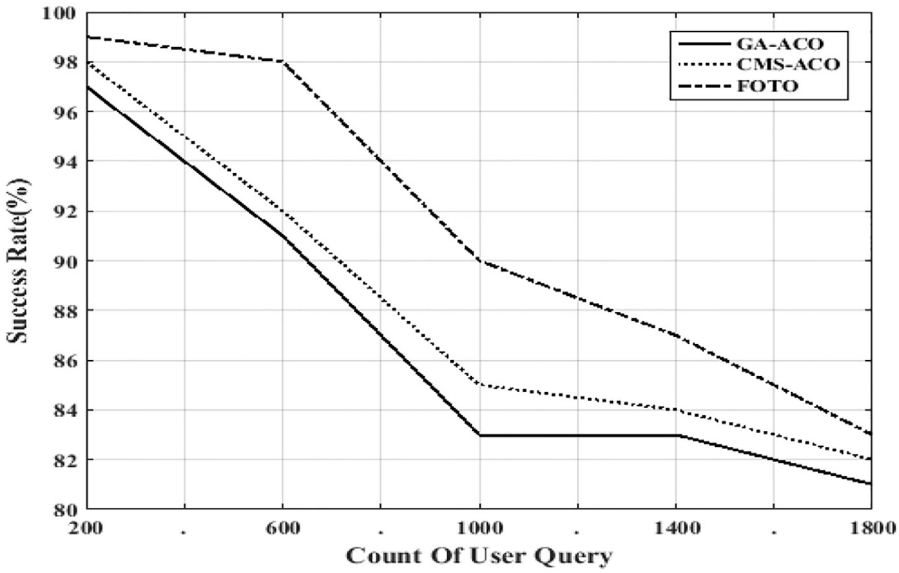


Fig. 10. Success Ratio.

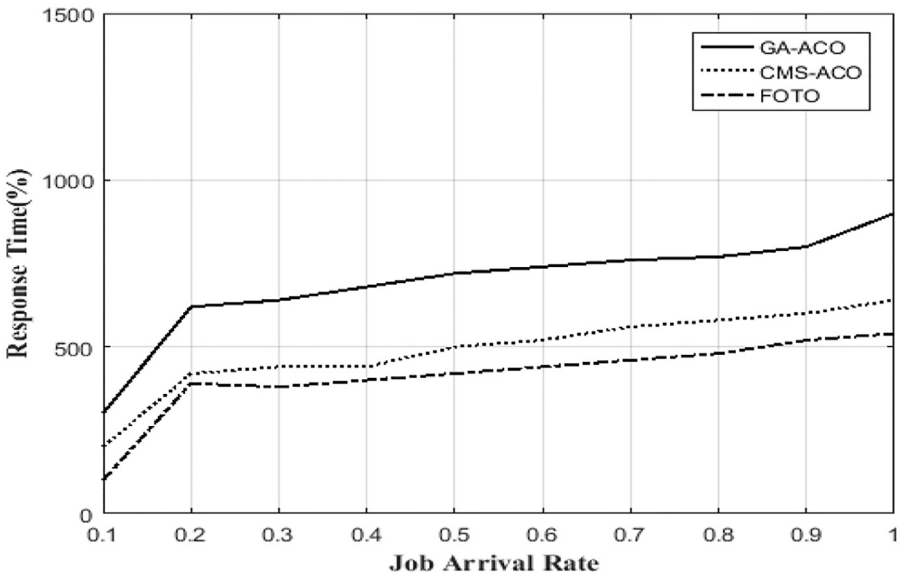


Fig. 11. Response Time Analysis.

by modifying the simulator source code written in Java language. The simulation graphs are taken from MatLab tool. The parameters used for the simulation are presented in Table 1.

5.1. Experimental results

This section represents the experiments and ends in the comparison of the proposed methodology. To show the effectiveness of our algorithm, the performances are compared with some other offloading algorithm such as CMS-ACO [1] and GA-ACO [23].

Fig. 3 represents the cost of the data centers and the experimental results shows that our proposed work is lower cost as compared to existing ones.

Fig. 4 demonstrates the resource utilization analysis for various requests. About 87% of utilization is achieved for the proposed algorithm as shown in figure. Resource allocation using the proposed optimization has improved the utilization of data center resources operating in dynamic environments. The task completion time is based on task scheduling complexity and resource allocation. When compared with the existing algorithm, our FOTO algorithm requires less computational time

as depicted by Fig. 5. The completion time is varied according to the job type. For each task, it performs some formulation and decisions hence, the proposed algorithm uses shorter time.

Figs. 6 and 7 explain the energy consumption of proposed FOTO scheme and the existing techniques. The energy consumption varies with the number of tasks. When the count of VMs is increased, the fruit fly optimization optimizes the resource and performs well in reducing the total energy consumption. The proposed FOTO scheme consumed 2.5 kWh energy while the GA-ACO and CMS-ACO have consumed 4.3 kWh and 6.5 kWh respectively.

The Figs. 8 and 9 represent the performance measures of the FOTO offloading method compared with the existing algorithm. FOTO uses multi-objective based optimum resource allocation for cloud data center. So, it finds the optimum host for task offloading. The response time and energy consumption of the FOTO are shorter and lower, respectively. The algorithm allocates the VM to a host provided that the required capacity is satisfied by the available resource capacity in the host. From the results, we observe that the our algorithm performs better among the existing algorithms, considering resource utilization, task completion time and energy consumption.

Figs. 10 and 11 describe the execution time analysis. The execution time is calculated by varying the number of cloudlets. While, execution time is reduced even using greater number of cloudlets. Fewer number of cloudlets require more execution time. The execution time required by FOTO approach is far better than the existing approaches.

6. Conclusion

We have proposed a novel FOTO algorithm which minimizes the energy consumption, lessens the task completing time and reduces the data center cost when offloading the tasks from smart devices to the cloudlet. The FOTO algorithm reformulated the user query and obtains the accepted requests utilizing linear regression model. Moreover, it has used multi-objective functions to find the optimum host in a cloud data center. The performance of the proposed algorithm has been evaluated by CloudSim 2.0 simulator and being compared with CMS-ACO and GA-ACO task offloading algorithms. The simulation results has justified that the FOTO algorithm guarantees lower energy consumption, less task completion time and lower data center cost, and it can be applied in smart health care systems using Edge-of-things computing. Resource allocation using the proposed optimization has improved the utilization of data center resources operating in dynamic environments and consumes less time and energy to complete the task. The execution time is calculated by varying the cloudlets number and it has been concluded that the time of proposed algorithm is far less than existing algorithms. In future, new fault tolerance mechanism will be designed to control the VM failures and more intricate offloading task relationship will be considered to further enhance energy saving for smart devices. In addition, security mechanism will also be applied for providing security protection of offloaded tasks.

Acknowledgment

This work was supported by the Fundamental Research Funds for the Central Universities under grant no. DUT16QY18.

References

[1] Wang, Tongxiang, Wei X, Tang C, Fan J. Efficient multi-tasks scheduling algorithm in mobile cloud computing with time constraints. Peer-to-Peer Netw Appl 2018;11(4). 793-807-15.
[2] Satyanarayanan, Mahadev, Bahl P, Caceres R, Davies N. The case for vm-based cloudlets in mobile computing. IEEE Pervasive Comput 2009;8(4):14–23.
[3] Lopez G, Pedro, Montresor A, Epema D, Datta A, Higashino T, Iamnitchi A, Barcellos M, Felber P, Riviere E. Edge-centric computing: vision and challenges. ACM SIGCOMM Comput Commun Rev 2015;45(5):37–42.
[4] Chen, Min, Qian Y, Hao Y, Li Y, Song J. Data-driven computing and caching in 5G networks: architecture and delay analysis. IEEE Wireless Commun 2018;25(1):70–5.
[5] Chen, Jing, He K, Yuan Q, Xue G, Du R, Wang L. Batch identification game model for invalid signatures in wireless mobile networks. IEEE Trans Mob Comput 2017;16(6):1530–43.
[6] Chen, Min, Hao Y, Li Y, Lai C-F, Wu D. On the computation offloading at ad hoc cloudlet: architecture and service modes. IEEE Commun Mag 2015;53(6):18–24.
[7] Zhou, Bowen, Dastjerdi AV, Calheiros RN, Srirama SN, Buyya R. A context sensitive offloading scheme for mobile cloud computing service. In: In cloud computing (CLOUD), IEEE 8th international conference on, IEEE; 2015. p. 869–76.
[8] Kumar, Karthik, Liu J, Lu Y-H, Bhargava B. A survey of computation offloading for mobile systems. Mobile Netw Appl 2013;18(1):129–40.
[9] Chen, Min, Hao Y, Qiu M, Song J, Wu D, Humar I. Mobility-aware caching and computation offloading in 5G ultradense cellular networks. Sensors 2016;16(7):974.
[10] Chen, Xu. Decentralized computation offloading game for mobile cloud computing. IEEE Trans Parallel Distrib Syst 2015;26(4):974–83.
[11] Chen, Min, Hao Y, Hu L, Huang K, Lau VKN. Green and mobility-aware caching in 5G networks. IEEE Trans Wireless Commun 2017;16(12):8347–61.
[12] Chunlin, Li, Yanpei L, Youlong L. Energy-aware cross-layer resource allocation in mobile cloud. Int J Commun Syst 2017;30(12).
[13] Zhang, Lei, Fu D, Liu J, Ngai EC-H, Zhu W. On energy-efficient offloading in mobile cloud for real-time video applications. IEEE Trans Circuits Syst Video Technol 2017;27(1):170–81.
[14] Zhang, Yin, Qiu M, Tsai C-W, Hassan MM, Alamri A. Health-CPS: healthcare cyber-physical system assisted by cloud and big data. IEEE Syst J 2017;11(1):88–95.
[15] Chen, Min, Zhang Y, Qiu M, Guizani N, Hao Y. SPHA: smart personal health advisor based on deep analytics. IEEE Commun Mag 2018;56(3):164–9.
[16] Al-Ayyoub, Mahmoud, Jararweh Y, Daraghmeh M, Althebyan Q. Multi-agent based dynamic resource provisioning and monitoring for cloud computing systems infrastructure. Cluster Comput 2015;18(2):919–32.
[17] Saqib, Fareena, Dutta A, Plusquellic J, Ortiz P, Pattichis MS. Pipelined decision tree classification accelerator implementation in FPGA (DT-CAIF). IEEE Trans Comput 2015;64(1):280–5.
[18] Yang, Shuo, Guo J-Z, Jin J-W. An improved id3 algorithm for medical data classification. Comput Electr Eng 2017.
[19] Xing, Bo, Gao W-J. Fruit fly optimization algorithm. In: In innovative computational intelligence: a rough guide to 134 clever algorithms. Cham: Springer; 2014. p. 167–70.

[20] Krishna, Venkata P. Honey bee behavior inspired load balancing of tasks in cloud computing environments. Appl Soft Comput 2013;13(5):2292–303.
[21] Li, Kun, Xu G, Zhao G, Dong Y, Wang D. Cloud task scheduling based on load balancing ant colony optimization. In: In Chinagrid conference (ChinaGrid), 2011 sixth annual. IEEE,; 2011. p. 3–9.
[22] Ramezani, Fahimeh, Lu J, Hussain FK. Task-based system load balancing in cloud computing using particle swarm optimization.? Int J Parallel Program 2014;42(5):739–54.
[23] Rashidi, Shima, Sharifian S. A hybrid heuristic queue based algorithm for task assignment in mobile cloud. Future Gener Comput Syst 2017;68:331–45.

Kai Lin is an associate professor at the School of Computer Science and Technology, Dalian University of Technology. His research interests include wireless communication, data mining and data fusion, big data analysis, mobile ad hoc networks, cyber physical systems, and sensor networks. He has authored or coauthored over 50 papers in international journals and conferences.

Sameer Pankaj is a student at the School of Computer Science and Technology, Dalian University of Technology. He is currently pursuing his M.S. degree, in Computer Science and Technology from Dalian University of Technology. His current research interests are Mobile Cloud Computing, Edge Computing, IoT.

Di Wang is a student at the School of Computer Science and Technology, Dalian University of Technology. He is currently pursuing his M.S. degree, in Computer Science and Technology from Dalian University of Technology. His current research interests are, 5G, IoT, network spectrum allocation.