EURASIP Journal on Wireless Communications and Networking

RESEARCH Open Access

Energy efficient edge-of-things

Asfa Toor[1], Saif ul Islam[2], Ghufran Ahmed[1], Sohail Jabbar[3], Shehzad Khalid[4] and Abdullahi Mohamud Sharif[5]*

Abstract

Edge-of-Things (EoT) emerged as a novel computing and storage paradigm to overcome the limitations of IoT-cloud environment by providing cloud-like services at edge of the network. EoT offers a vast area for research and development as the invention has laid out great opportunities to experiment the possibilities for handling large data sets produced by the growing Internet-of-Things (IoT). The EoT offers a framework that lies between the cloud-to-end to perform the processing and cater the storage demands of the IoT applications. However, the exponential increase in EoT infrastructure resulted into extreme energy consumption. This paper finds the opportunity to address the issue of energy consumption in IoT-EoT environment by introducing dynamic speed scaling mechanism in EoT devices. The proposed approach is rigorously evaluated, and the verification is acquired through the simulations carried out on the simulator, iFogSim. The results show significant improvement in energy conservation by dynamically scaling the processor frequency of EoT devices according to the load variations in IoT traffic.

Keywords: Edge-of-things, Fog computing, Energy management, Dynamic speed controller, iFogSim

1 Introduction

Edge-of-Things (EoT) has emerged in the progressing world of information technology, as a paradigm liable to provide utmost quality of processing and storage services at edge of the network. The Internet of Things (IoT) is progressing rapidly offering an infrastructure that is most viable to bring a significant quality enhancement to human life and productivity [1]. However, the growing advancements in this technology demands a platform that is more energy-efficient and causes lesser delay. The need to process and store huge volumes of data at a fast pace is the major concern for leveraging resources. IoT devices backbone has two major infrastructures, cloud and fog/edge computing. Both of them significantly provide efficient data storage and resource management, enabling the developers to create interactive Internet service platforms for smart access.

Cloud is a large cluster of isolated servers that provide Infrastructure as a service (IaaS) together, allowing various clients and organizations to rent the service for storage and computing services over the Internet [2]. Cloud computing can provide scalable systems and on-demand processing but it is not a good option for health monitoring and emergency response system due to delay in transmission.

Edge computing was introduced as the solution of bottle neck on cloud by placing computing resources near sensors. However, the edge devices are unable to handle numerous applications that demand limited resources because it causes resource conflict and huge processing latency.

EoT/fog computing on the other hand impeccably combines edge devices and cloud resources to help overcome these limitations. Fog takes a complimentary lead compared to cloud it comes to low latency and high throughput since it is much closer to the edge of the network. Fog computing is a novel approach for achieving less delay by extending cloud services to the edge [3, 4].

The EoT is an intermediate layer between the edge of the network and cloud. It underlies Mega cloud and its function is to filter and pre-process the data before sending it to cloud [5]. Cloud being the parent to EoT/fog is huge and top tier in this pyramid faces a lot of issues; including less throughput and more delay in the services, as compared to fog which also inherits some of these major issues from cloud. The paper discusses the issue of energy consumption in fog devices which also lies in the major challenges category but still has not received much attention.

*Correspondence: abdullahi.shariif@uniso.edu.so
[5]University of Somalia, Mogadishu, Somalia
Full list of author information is available at the end of the article

To streamline the increased demands of large IoT applications, more EoT/fog devices need to be deployed that ultimately results in higher consumption of energy and performance issues [6]. Similarly, the inefficient utilization of EoT/fog-IoT resources also cause a tremendous increase in energy consumption which certainly invokes the need to develop techniques liable for managing power consumption with ensuring quality of service guaranteed by the EoT/fog computing infrastructure [7]. The EoT/fog-IoT computing infrastructure carries several issues and challenges [5]. However, the resource constrained nature of EoT/fog-IoT nodes makes energy a most significant challenge to be raised on the agenda [8]. Hence, the paper aims to achieve energy efficiency in EoT/fog computing by a proposed methodology, which has proven to be much efficient than any other existing solutions.

The proposed methodology involves the dynamic speed scaling of the CPU based upon the amount of workload being subjected upon the application. The processor speed scaling is a technique been implemented by many researchers for gaining lower rates of energy consumption. The increased speed of processor causes higher rates of energy consumption and a processor executing on lower speed consumes energy on a comparatively lower rate. However, excessive and frequent switching of processor speed increases the temperature of microprocessors, which causes unreliability of the system [9, 10].

Hence, the proposed system maintains a counter which monitors the number of iterations a particular type of workload (higher, lower, or moderate) has been encountered and then takes a run-time decision that whether the system should alter the speed of the processor or not. If the speed is to be altered then at which particular speed the processor shall execute the commands upon is assessed according to the workload subjected upon the system.

To the best of our knowledge, the proposition made by the authors is the first effort that implements a dynamic framework to alter processor speed for gaining energy efficiency in fog infrastructure. The simulations are carried out in the simulator, iFogSim, and the results gained manifest a promising solution for achieving an energy-efficient system.

The terms EoT and fog are used interchangeably by the authors as both the infrastructures form an intermediary layer between the cloud and edge of the network and offer similar services.

The main contributions of the paper are manifold:

1. The dynamic framework to achieve energy efficiency in fog computing.
2. The maintenance of counter checks to avoid frequent change of processor speed.
3. The integration of the proposed methodology upon an IoT-fog infrastructure.
4. The extensive simulations carried out and tested of the proposed system in the simulator, iFogSim.
5. The testings carried out upon all the types of possible workloads; lower, moderate, and higher workloads.

The rest of the paper is organized as follows. Section 2 presents the related work carried out by other researchers to overcome the issue of power consumption in fog computing. The proposed system architecture is described in Section 3 followed by Section 4 presenting the proposed methodology. Section 5 is comprised of experimental setup that describes the simulation environment for carrying out the implementation of proposed algorithm and results. Finally the conclusions and future directions are explained in Section 6.

2 Related work

The authors in [7, 11, 12] offer a wider and detailed view of fog computing where they present it as a set of technologies that have been merged together to be developed and matured in a distinct manner. The wide span of fog computing can be assessed by a single integration of information technology (IT) scenario in fog computing which can lead to new hybrid possibilities and requirements. The fog is advancing, but in a good way it is shifting the cloud computing and merging it with edge computing for making a liable platform of fog computing where various applications are being developed, i.e., the IoT applications. The paper highlights the set of challenges that fog computing faces are the issues relating to security, management, standardizations, accountability/monetization, and programmability.

Another study [4, 13] shows the dynamic infrastructure of fog computing for data transmission and processing with minimum service delay. The authors brought about a smart gateway, for bridging fog with cloud, creates a priority queue that performs dynamic demand side management. The queue is affected by not only the consumer importance but also the consumer policies and the status of energy resources. They discussed the load-shedding approach thoroughly and declared that a micro-grid determines the load shedding based upon the battery level and priority queue. Furthermore, for enhancing performance and data transmission time, fog computing being the middle layer between IoT applications and cloud has emerged as a promising paradigm.

For overcoming the crisis of energy management, the authors of [14] have implemented two platforms which are scalable, flexible, and open-source software/hardware comprising platforms. Energy management in special domains is required by the evolution of microgrids, its management, and generation is becoming a major question now a days as it supports to grasp zero net energy. The two proposed prototypes are HEM and micro grid-level, and they implement energy management-as-a-service on fog computing. For residential and commercial use, the microgrid is seen to show improve trustworthiness, proficiency, and viability.

Another study [15, 16] have proposed energy-efficient fog servers which deliver the information to mobile users (in vehicle). They used renewable energy, non-renewable energy, and also the combination of both. The research not only brought energy saving but also reduced carbon footprint. The paper explains that smart cities utilize smart grids, intelligent energy management, smart signaling, mobile infotainment, traffic management, and etc. The integration of the microgrids with fog can reduce energy consumption of the IoT applications in vast ways. It shows that in such a system the system can work more efficiently irrespective of the other three parameters because of the availability of the renewable energy.

The integration of microgrids with the fog is proposed and implemented in [17, 18] for utilizing the energy consumption of the IoT devices. The solution integrates the microgrids in the IoT gateway in which the gateway is capable of taking a decision that whether it should send the data for processing and computation to the cloud or should it compute it locally saving the power consumption. The authors claimed that research in learning and exploring the machine learning ways to attribute the IoT gateways for dealing with the automatic decision-making that whether the data should be sent to the local devices or should be transmitted to the cloud, should be taken into account. Incorporating machine learning techniques and making cognitive IoT gateways can enhance the performance of IoT applications and hence should be pondered upon. Also, designing an application suitable for consuming lesser energy has been suggested by the authors.

Furthermore, the power consumption issues in fog and cloud computing have been explored thoroughly, and the reasons behind higher power consumptions in fog computing as compared to cloud have been highlighted. The authors formulated a workload allocation problem which suggested the optimal workload allocations between fog and cloud providing the minimal power consumption along with a constrained service delay. Calculations and the result showed that the cloud computing is more powerful and energy-efficient than fog computing. However, the fog gives us the advantage of physical proximity providing the better performance and services [19].

In contrast to the study made in [19, 20] gives a detailed comparison of energy consumed by an application running on centralized data centers (DCs) in cloud computing and the energy consumed by the application running on the nano data centers (nDCs) used in fog computing and give quite a distinctive result. The research found that the nDCs "might" lead to energy-saving depending on the factors; type of access network attached to nano servers, the ratio of active time to idle time, type of application, and number of data pre-loading. Also that the nDCs would work best (in terms of energy efficiency) if the data generated at end-user premises is larger, which is not very frequent and common.

Dynamic voltage and frequency scaling (DVFS) is a solution applied by the researchers to attain energy efficiency in cloud computing where they performed the testings in a simulator called CloudSim [21]. As the name suggests, it is an adjustment of the processors or controller chips in a computer system or device in order to increase its speed and energy efficiency. In the Linux Kernel, DVFS is performed on five modes, which are Performance, PowerSave, UserSpace, Conservative, and OnDemand modes.

For testing varying environments and applications, an ultimate solution for testing fog computing advantages and its constraints in a real-time environment is presented in [22]. The authors introduce a simulator, "iFogSim," which enables the simulation of resource management and application scheduling policies across cloud and edge resources under different scenarios and conditions. Also, the researchers enforced the research to be made upon the biggest challenge that most fog computing solution face, i.e., power-aware resource management policies. They illustrated the fact that "how to get extra battery life for fog device" being the biggest challenge and bringing about future directions for researches of the field to look into new policies that base upon battery life of devices.

The [21] explains the five modes for activating DVFS with varying configurations in a cloud. These modes are *Performance, Conservative, PowerSaver, UserSpace*, and *OnDemand*. The first three use fixed frequencies or execute on a same speed; however, the last two are dynamic. OnDemand mode being one of the dynamic frequency-altering modes executes upon a higher frequency until it receives the lower workload from an application for a certain amount of time, then the OnDemand switches to a lower speed of CPU.

The above discussed work is related to our work in different aspects like energy conservation, simulating EoT environment, and frequency scaling; however, up to our knowledge, we are the first to introduce dynamic speed scaling mechanism to achieve energy efficiency in EoT environment .

3 Methodology

3.1 Proposed system architecture

The system architecture of the proposed methodology targets the energy consumption of the EoT/fog device, which lies in between the IoT applications and the cloud. The IoT applications can comprise of any possible mode of communication [23] like a phone, a surveillance camera, a watch, laptop, a car, or a television with help of sensors and actuators as illustrated in the Fig. 1. The IoT applications use EoT/fog device to address its urgent processing and storage demands which can cause extreme energy consumption especially under higher workloads. The EoT/fog devices by default execute in a performance mode (without onDemand mode) where the CPU runs on the maximum speed for providing up-to-the-mark service qualities without considering the catch that it would generate higher energy consumption. The proposed scheme implements a dynamic speed scaling upon the EoT/fog device through a dynamic speed controller which alters the CPU speed so that the system shifts to a lower CPU speed under specified conditions in order to gain maximum utilization of the energy while ensuring the same quality of service.

The dynamic speed controller monitors the workload of the input application (IoT application) and takes a prompt decision that whether it should increase the CPU speed or decrease it to yield minimum energy consumption. The dynamic speed controller keeps track of the amount of workload being subjected upon the EoT/fog device by the IoT applications, in a way that it alters the speed of the CPU under specified conditions. This track is maintained by the counters which are the part of algorithm that determines the requirement for CPU speed alteration. Frequent CPU speed variations are avoided as this increases the chance of an overheated microprocessor which results in a system failure [9, 10]. The conditions under which the speed is altered are discussed in detail in Section 3.3

Before understanding the algorithm and exact working of the system, it is important to know the application upon which the methodology is tested and experimented. Hence, the next section describes in detail the IoT application, its design and composition, and the details of the simulator in which the environment is created.

3.2 Case study

The IoT application designed for executing the methodology is a scenario of a *security surveillance camera* which receives live video streams and are attached to a PTZ controller for monitoring areas that indicate threat [22]. The processed video streams that indicates a motion is sent to the gateway; the fog device in the case manages the activity of the cameras. The fog devices are attached to four cameras per area. The number of areas indicate the number of fog devices in the physical topology of the simulation environment and is increased as the number of areas to be surveilled increase. Here, in the topology, the higher workload is signaled by the number of areas the system aims to surveil.

The application has five major modules motion detector, object detector, object Tracker, pan-tilt-zoom (PTZ) control, and user interface. The cameras attached have the PTZ control that detect the activity in the area and catches the video stream of the motion by adjusting the camera through the controllers. All the five modules reside in the surveillance cameras that help capturing the video indicating threat and are controlled by the gateway attached, i.e., the fog device managing four cameras per area [22].

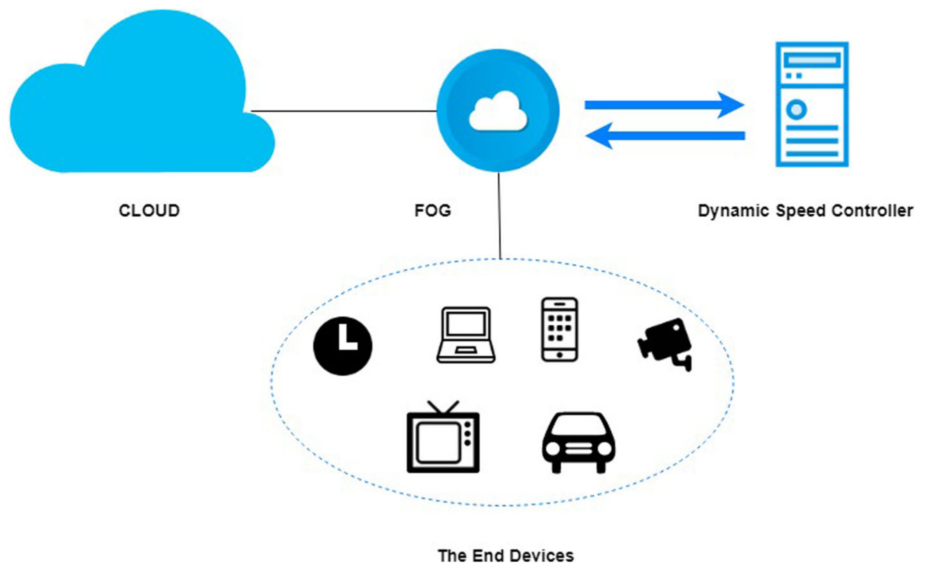

Fig. 1 System architecture

The number of areas or the amount of workload is altered between 1 and 16, as for a higher number the application running in the simulator would generate an "out-of-memory" error for different level of CPU speeds. Hence, the methodology is tested for varying amount of areas between the range 1 to 16.

3.3 Algorithms

The algorithm for the implementation of the OnDemand mode comprises of three stages. The amount of workload being subjected upon the IoT application is monitored at each iteration so as to keep a track of number of times a lower, higher, or a moderate amount of workload is passed to the system.

Three counters, *CounterA*, *CounterB*, and *CounterC* are maintained which are designed for managing the three different workload categories (lower, higher, and moderate). The counter function indicates the need to switch to a varied CPU speed or to retain the current CPU speed upon which the EoT/fog devices are currently executing the demands of IoT applications. The capacity of each counter is set to 2, if the amount of a specified workload category is reached up to 2 iterations then the CPU speed is changed; otherwise, it is left unaltered. The purpose for designing the counter function was to avoid frequent CPU speed alterations.

3.3.1 Allocating areas to counter

The first stage of the methodology is illustrated in Algorithm 1 where the workload is passed by an IoT application to the EoT/fog device for processing. This data is then passed to the respective counter function. The workload is the number of areas being subjected by the application for monitoring and surveillance. The categories of the workload is divided to three, as explained in Table 1. The lower workload is assigned to CounterA function, the moderate is assigned to CounterB function, and the higher workload is allocated to the CounterC function.

3.3.2 The counter function

The second stage of the methodology invokes the respective counter function that monitors the particular workload category for up to two iterations. The Algorithm 2 illustrates a for loop that runs for two iterations and adds the workload values in an array. Once the for loop is ended, the outer If loop checks the value of the array, if it lies below 2 (for lower category case) then it sends a signal to the dynamic speed scaling function which alters the CPU speed accordingly.

Here in the paper, the function for CounterA is explained only because CounterB and CounterC functions would only change the range specified in the line 6 of Algorithm 2.

Algorithm 1 Allocating areas to counter

```
procedure EnergyManagement($areas$)
    if $areas \geq 1$ and $areas \leq 4$ then
        $CounterA \leftarrow areas$
    else
        if $areas \geq 5$ and $areas \leq 8$ then
            $CounterB \leftarrow areas$
        else
            if $areas \geq 9$ and $areas \leq 16$ then
                $CounterC \leftarrow areas$
            end if
        end if
    end if
end procedure
```

Table 1 The workload categories

Categories	Number of areas
Lower	1–4
Moderate	5–8
Higher	9–16

3.3.3 Dynamic speed scaling

The third and the final stage of the proposed methodology alters the speed of the CPU according to the received signal from the counter function of second stage. Here, if the CounterA function has received a signal that equals to the value 2, then the CPU speed (million instructions per sec (MIPS)) of the fog device would be decreased to 28.6% and also the MIPS of cameras attached would be decreased to 90% . In case the signal received invokes CounterB then the value will remain unchanged, which is the default system value. When CounterC is invoked, then the system runs at its highest set CPU speed which is 28.6% higher from the lowest CPU speed which is illustrated in the Algorithm 3.

Algorithm 2 The counter function

```
1:  procedure CounterFunction($areas$)
2:      for $i = 0$, $i{+}{+}$, while $i \leq 2$ do
3:          $myArray = areas$
4:          $i{+}{+}$
5:      end for
6:      if $myArray \geq 0$ and $myArray \leq 5$ then
7:          return 2
8:      else
9:          return 0
10:     end if
11: end procedure
```

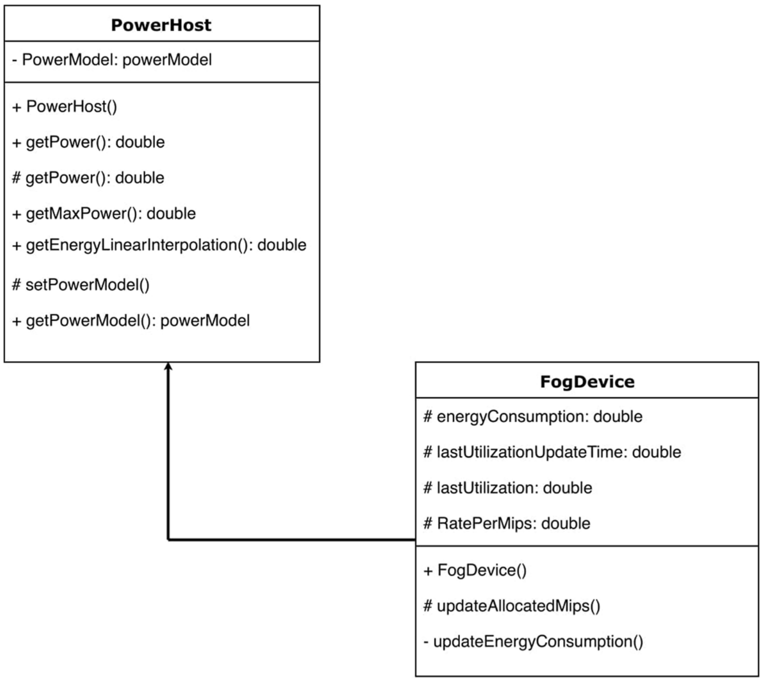

Fig. 2 Class diagram for energy calculation

The designing of such an application that is liable to consume lesser energy than any conventional IoT application is emphasized by the authors of the paper [17]. In the methodology proposed, the default system configurations and the lowest CPU speed configurations run on such an application that is most efficient in terms of energy consumption. However, the third mode where the system receives higher workload from the IoT application is set to performance level configuration where it consumes a little higher level of energy but is most effective for performing desired computation and processing demands.

Algorithm 3 Dynamic speed scaling

```
1:  procedure SPEEDSCALING($CounterA, CounterB, CounterC$)
2:      if $CounterA = 2$ then
3:          $MIPSvalue \leftarrow decreased$
4:      else
5:          if $CounterB = 2$ then
6:              $MIPSvalue \leftarrow default$
7:          else
8:              if $CounterC = 2$ then
9:                  $MIPSvalue \leftarrow increased$
10:             end if
11:         end if
12:     end if
13: end procedure
```

4 Experimental section

This section presents the simulation environment, and results and discussion.

4.1 Simulation environment

Implementation of the proposed methodology is done in the simulator called iFogSim [22] which provides a platform for implementing a dynamic environment of an IoT application. For evaluation of the challenges faced in the related field (fog and IoT), a real environment is desired but that is extremely costly and does not provide repeatable environments for carrying out tests. Therefore, the iFogSim enables the simulation of resource management and application scheduling policies across cloud and edge resources under different scenarios and conditions. The application model of iFogSim is sense-process-actuate in which sensors publishes data to IoT networks, the data is processed by applications running on fog devices and then forwarded to the actuators. The architecture of iFogSim is composed upon the basic event simulation functionalities of CloudSim. iFogSim has five major *classes*, i.e., fogdevice, sensor, tuple, actuator, and application. In addition to that, the simulated services available in iFogSim are power monitoring service and resource management service.

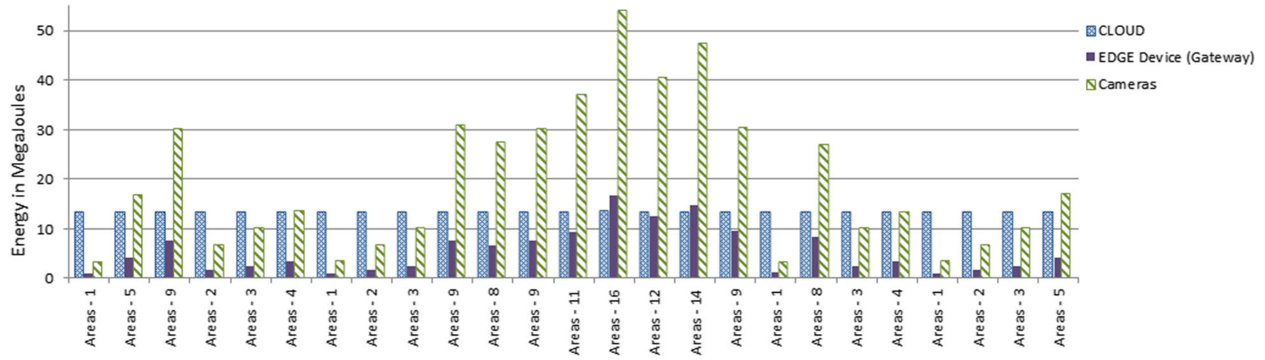

Fig. 3 The energy consumption of the system

iFogSim supports simulations on the scale expected in the context of IoT.

4.1.1 *The energy calculation function*

The class fogdevice incorporates a function for calculating energy consumption where the key parameters are time and MIPS utilization value. The function rolls back to the CloudSim class called *PowerHost* in which the power model is formulated by calculation of the energy consumption using linear interpolation of the utilization change through the function called *getEnergyLinearInterpolation()*.

Table 2 The inputs and system response

Input	Function invoked	CPU speed
Areas - 1	CounterA	Default
Areas - 5	CounterB	Default
Areas - 9	CounterC	Default
Areas - 2	CounterA	Default
Areas - 3	CounterA	Default
Areas - 4	CounterA	Lowest
Areas - 1	CounterA	Lowest
Areas - 2	CounterA	Lowest
Areas - 3	CounterA	Lowest
Areas - 9	CounterC	Lowest
Areas - 8	CounterB	Lowest
Areas - 9	CounterC	Default
Areas - 11	CounterC	Highest
Areas - 16	CounterC	Highest
Areas - 12	CounterC	Highest
Areas - 14	CounterC	Highest
Areas - 9	CounterC	Highest
Areas - 1	CounterC	Highest
Areas - 8	CounterB	Default
Areas - 3	CounterA	Default
Areas - 4	CounterA	Lowest
Areas - 1	CounterA	Lowest
Areas - 2	CounterA	Lowest
Areas - 3	CounterA	Lowest
Areas - 5	CounterC	Lowest

Figure 2 depicts the two classes responsible for calculating energy. Also, it is to be noted that the demonstrated class fogdevice, illustrates only the concerned functions required for calculating energy and does not include details of other functions and parameters of the class.

4.2 Results and discussion

This section provides the evaluation of proposed technique in terms of energy consumption, network usage, execution time, and loop delays.

4.2.1 *Energy consumption*

The energy consumption of the system is set forth in the Fig. 3 where it is clear that the energy consumption shows a significant increase when the system executes at the maximum speed or the highest value of MIPS.

The inputs given to achieve the results and the system response in terms of the counter function it invoked along with the CPU speed set by the proposed methodology is set forth in Table 2. Here, it is to be noticed that the system waits for at least two values before it shifts to another counter. This is because frequent CPU speed alteration is not recommended as it causes system unreliability and inefficiency.

The comparison of the energy consumption of the system executing in OnDemand mode with the system that executes completely on fog and does not perform any kind of speed scaling is illustrated in the Fig. 4. The results show a significant gain in the energy utilization as the OnDemand mode exhibits a drastic transformation in the graph when the speed scaling is performed. However, the case where the speed scaling is set to the highest, i.e., when the workload is maximum, only in that case the graph comes aligned to the graph of system executing without the OnDemand mode.

4.2.2 *Network usage*

Figure 5 shows that the system consumes a significantly lower network bandwidth under dynamic alteration of the CPU speed. The graph however aligns with

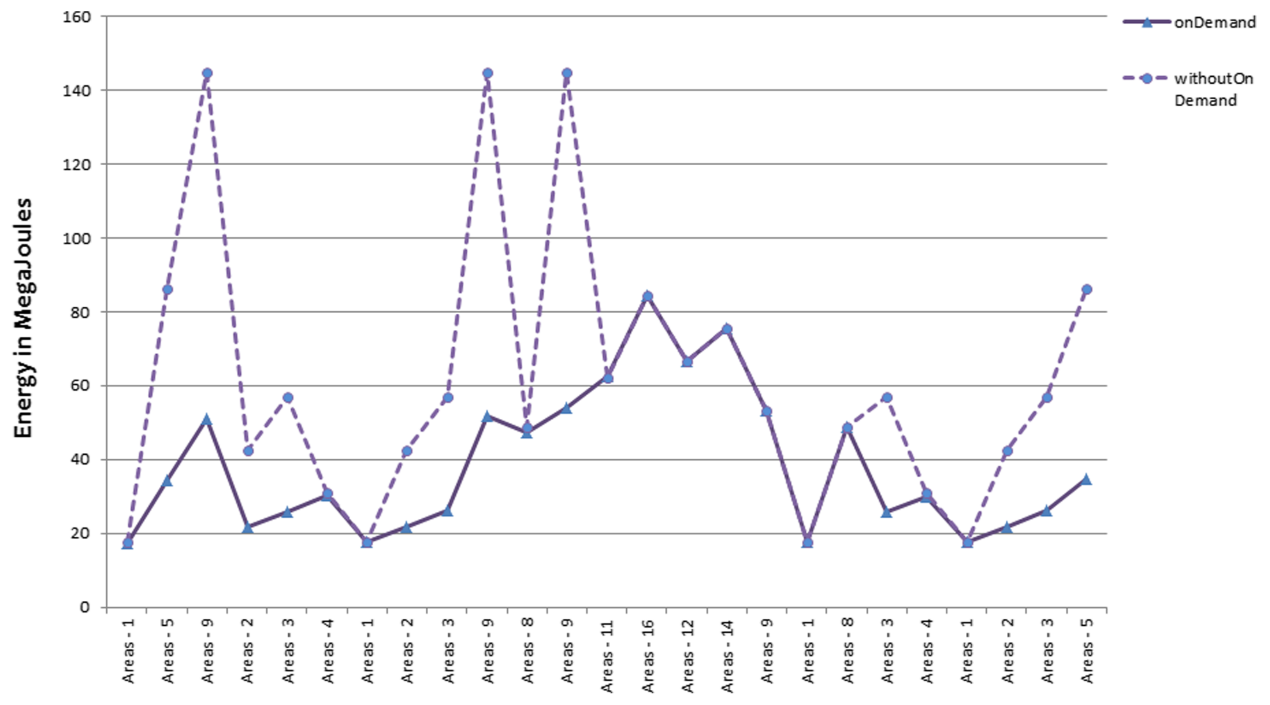

Fig. 4 The energy consumption of OnDemand mode versus without the OnDemand mode

the "without OnDemand" bar when the CPU speed is set to the highest mode for increased amount of workload scenario. Other than that, the network usage has also produced credible results for the OnDemand mode.

4.2.3 Execution time

The execution time of the OnDemand mode shows a promising result in achieving minimum time for processing the demands of IoT applications. As shown in the Fig. 6 the execution time of OnDemand mode merges

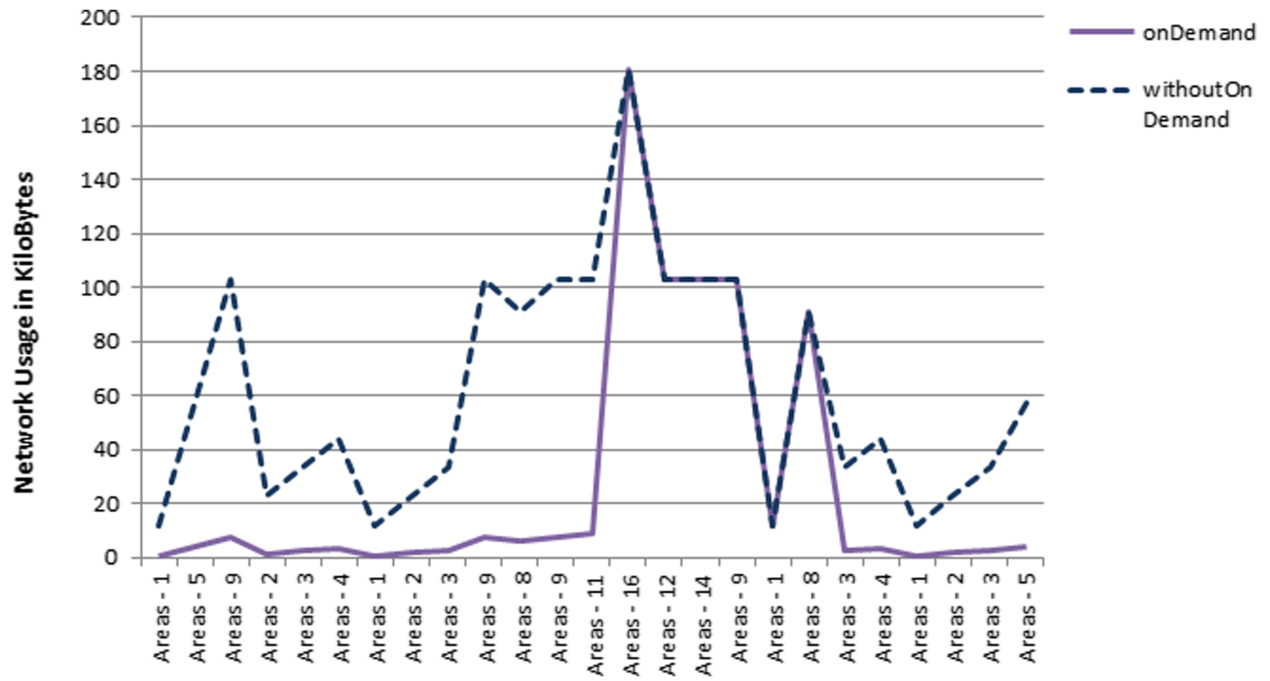

Fig. 5 The network usage of the OnDemand mode versus without the OnDemand mode

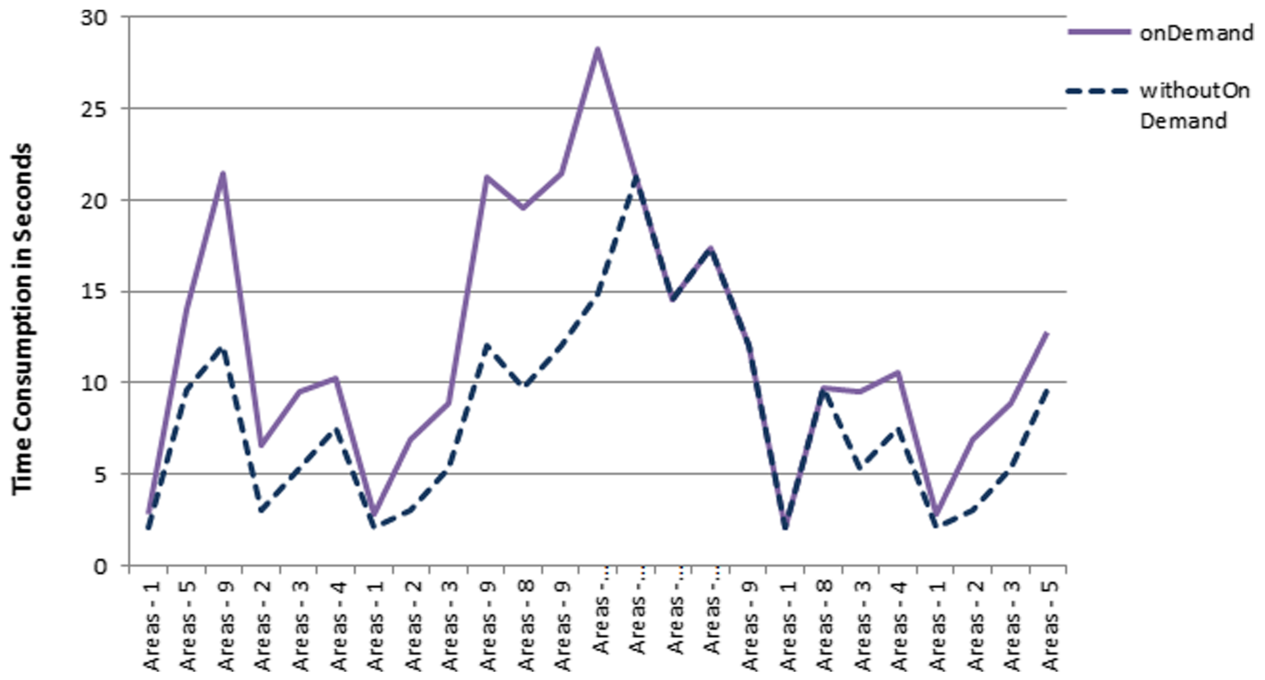

Fig. 6 The execution time of the OnDemand mode versus without the OnDemand mode

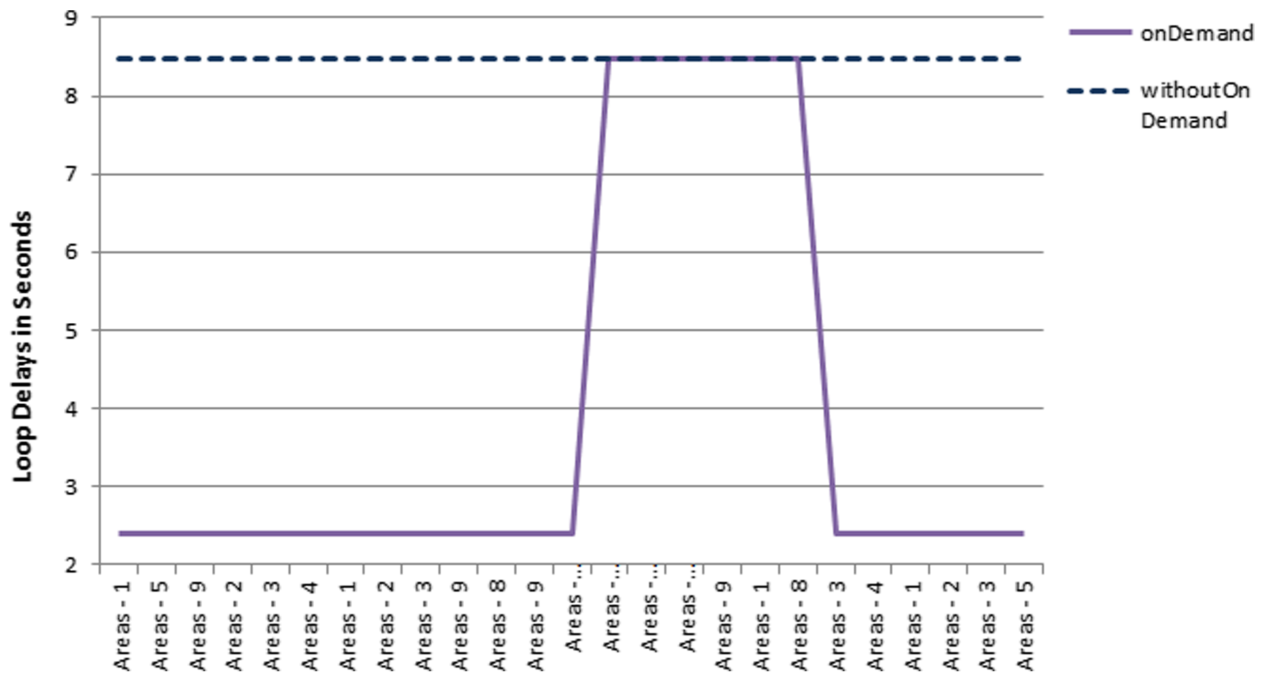

Fig. 7 The loop delays of the OnDemand mode versus without the OnDemand mode

with the correspondent graph only when the CPU speed is set to the highest mode. Otherwise, even for higher configurations, the OnDemand gives a significantly better result.

4.2.4 Loop delays

The inner loop delays of the IoT application that are greatly responsible for assessing the better service quality of fog computing over cloud computing is showing a drastic difference between the two comparison graphs (Fig. 7). The graph of without the OnDemand mode shows a linearity at about 8.45 s whereas the OnDemand mode gives the loop delay at approximately about 2.45 s which is a drastic drop of 71% from 8.45 s. However, again both the graphs converge when the CPU speed is set to the highest configurations and consume equal loop delays.

5 Conclusion

The paper has proposed a framework to extend the possibilities of advancement in EoT infrastructure with least energy consumption. The authors have targeted a dynamic frequency scaling technique to gain maximum efficiency in energy consumption of the EoT/fog devices and IoT applications. The methodology has proven to be immensely liable for utilization of least energy while assuring maximum quality of service. The frequency scaling is controlled dynamically where the system monitors the workload through three counters and performs the scaling when a uniformity in the amount of workload is encountered which makes the system prone to resist frequent alteration of its configurations. The authors have resisted the frequent speed scaling as it harms the system performance and hence the proposed methodology has proven to be the most appropriate solution to generate minimum consumption of energy even under higher processing or storage demands by an IoT application. The future work aims at targeting the possibilities to integrate the mechanisms and framework of machine learning in the EoT/fog-IoT infrastructure where the techniques of machine learning can help predict the future processing demands of an IoT application based upon the previous encountered behavior of the application.

Abbreviations

DCs: Data centers; DVFS: Dynamic voltage and frequency scaling; EoG: Edge of Things; HEM: Home energy management; IAAS: Infrastructure as a service; IOT: Internet of Things; IT: Information technology; MIPS: Million instructions per second; nDCs: Nano data centers; PTZ: Pan-tilt-zoom

Acknowledgements

Not Applicable.

Funding

This work does not have any funding.

Availability of data and materials

Data sharing not applicable to this article as no datasets were generated or analyzed during the current study.

Authors' contributions

AT and SuI conceived the main idea, designed the algorithm, and proposed an EoT-IoT framework. AT performed the implementation of the proposition in the iFogSim simulator. GA, SJ, SK, and AMS contributed to the structuring, reviewing, and finalizing of the manuscript. All authors read and approved the final manuscript.

Competing interests

The authors declare that they have no competing interests.

Publisher's Note

Author details

[1]COMSATS University, Islamabad, Pakistan. [2]Dr. A. Q. Khan Institute of Computer Sciences and Information Technology, Kahuta, Pakistan. [3]National Textile Universities, Faisalabad, Pakistan. [4]Department of Computer Engineering Bahria University, Islamabad, Pakistan. [5]University of Somalia, Mogadishu, Somalia.

Received: 11 July 2018 Accepted: 5 March 2019
Published online: 29 March 2019

References

1. M. Abdel-Basset, G. Manogaran, M. Mohamed, E. Rushdy, in *Concurrency and Computation: Practice and Experience*. Internet of things in smart education environment: Supportive framework in the decision-making process (Wiley Online Library, 2018), p. e4515
2. S. Aslam, S. ul Islam, A. Khan, M. Ahmed, A. Akhundzada, M. K. Khan, Information collection centric techniques for cloud resource management: Taxonomy, analysis and challenges. J. Netw. Comput. Appl. **100**, 80–94 (2017)
3. Y. N. Krishnan, C. N. Bhagwat, A. P. Utpat, in *2015 2nd International Conference on Electronics and Communication Systems (ICECS)*. Fog computing—network based cloud computing *Electronics and Communication Systems (ICECS)* (IEEE, 2015), pp. 250–51
4. A. V. Dastjerdi, H. Gupta, R. N. Calheiros, S. K. Ghosh, R. Buyya, in *Internet of Things*. Fog computing: principles, architectures, and applications (Elsevier, 2016), pp. 61–75
5. M. Aazam, M. St-Hilaire, C.-H. Lung, I. Lambadaris, in *2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC)*. Pre-fog: IoT trace based probabilistic resource estimation at fog (IEEE, 2016), pp. 12–17
6. V. D. Reddy, B. Setz, G. S. V. Rao, G. Gangadharan, M. Aiello, Metrics for sustainable data centers. IEEE Trans. Sustain. Comput. **2**(3), 290–303 (2017)
7. Z. Hao, E. Novak, S. Yi, Q. Li, Challenges and software architecture for fog computing. IEEE Internet Comput. **21**(2), 44–53 (2017)
8. K. Kaur, T. Dhand, N. Kumar, S. Zeadally, Container-as-a-service at the edge: Trade-off between energy efficiency and service availability at fog nano data centers. IEEE Wirel. Commun. **24**(3), 48–56 (2017)
9. R. Fahrizal, R. Alfanz, A. Sakti, in *2016 4th International Conference on Cyber and IT Service Management*. Analysis of effect overclocking durability on intel processor (IEEE, 2016), pp. 1–4
10. H. B. Jang, J. Lee, J. Kong, T. Suh, S. W. Chung, Leveraging process variation for performance and energy: In the perspective of overclocking. IEEE Trans. Comput. **63**(5), 1316–1322 (2014)
11. L.M. Vaquero, L. Rodero-Merino, Finding your way in the fog: towards a comprehensive definition of fog computing. ACM SIGCOMM Comput. Commun. Rev. **44**(5), 27–32 (2014)
12. M. Aazam, E.-N. Huh, in *2015 IEEE 29th International Conference on Advanced Information Networking and Applications*. Fog computing micro datacenter based dynamic resource estimation and pricing model for IoT (IEEE, 2015), pp. 687–694
13. K. Shahryari, A. Anvari-Moghaddam, in *2017 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*. Demand side management using the internet of energy based on fog and cloud computing (IEEE, 2017), pp. 931–36
14. M. A. Al Faruque, K. Vatanparvar, Energy management-as-a-service over fog computing platform. IEEE Internet Things J. **3**(2), 161–169 (2016)

15. S. Igder, S. Bhattacharya, J. M. Elmirghani, in *2016 10th International Conference on Next Generation Mobile Applications, Security and Technologies (NGMAST)*. Energy efficient fog servers for Internet of Things information piece delivery (IoTIPD) in a smart city vehicular environment (IEEE, 2016), pp. 99–104
16. F. Jalali, S. Khodadustan, C. Gray, K. Hinton, F. Suits, in *2017 IEEE International Conference on Edge Computing (EDGE)*. Greening IoT with fog: a survey (IEEE, 2017), pp. 25–31
17. F. Jalali, A. Vishwanath, J. de Hoog, F. Suits, in *2016 IEEE Innovative Smart Grid Technologies-Asia (ISGT-Asia)*. Interconnecting fog computing and microgrids for greening IoT (IEEE, 2016), pp. 693–698
18. F. Jalali, O. J. Smith, T. Lynar, F. Suits, in *Proceedings of the SIGCOMM Posters and Demos*. Cognitive IoT gateways: automatic task sharing and switching between cloud and edge/fog computing (ACM, 2017), pp. 121–123
19. R. Deng, R. Lu, C. Lai, T. H. Luan, H. Liang, Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption. IEEE Internet Things J. **3**(6), 1171–1181 (2016)
20. F. Jalali, K. Hinton, R. Ayre, T. Alpcan, R. S. Tucker, Fog computing may help to save energy in cloud computing. IEEE J. Sel. Areas Commun. **34**(5), 1728–1739 (2016)
21. T. Guérout, T. Monteil, G. Da Costa, R. N. Calheiros, R. Buyya, M. Alexandru, Energy-aware simulation with dvfs. Simul. Model. Pract. Theory. **39**, 76–91 (2013)
22. H. Gupta, A. Vahid Dastjerdi, S. K. Ghosh, R. Buyya, ifogsim: a toolkit for modeling and simulation of resource management techniques in the internet of things, edge and fog computing environments. Softw. Pract. Experience. **47**(9), 1275–1296 (2017)
23. M. Abdel-Basset, G. Manogaran, M. Mohamed, Internet of things (IoT) and its impact on supply chain: a framework for building smart, secure and efficient systems. Futur. Gener. Comput. Syst. **86**, 614–28 (2018)