Proceedings of the 12th INDIACom; INDIACom-2018
5th 2018 International Conference on "Computing for Sustainable Global Development", 14th – 16th March, 2018
Bharati Vidyapeeth's Institute of Computer Applications and Management (BVICAM), New Delhi (INDIA)

Appropriate Deployment of Fog Nodes for IoT Services

Prasenjit Maiti
Department of Computer Science and Engineering
National Institute of Technology, Rourkela
Email: pmaiti1287@gmail.com

Jaya Shukla
Department of Computer Science and Engineering
National Institute of Technology, Rourkela
Email: jayashukla2192@gmail.com

Bibhudatta Sahoo
Department of Computer Science and Engineering
National Institute of Technology, Rourkela
Email: bibhudatta.sahoo@gmail.com

Ashok Kumar Turuk
Department of Computer Science and Engineering
National Institute of Technology, Rourkela
Email: akturuk@nitrkl.ac.in

Abstract –
The Fog computing technology has come out as a feasible option to reduce the gap between cloud applications and the the physical IoT devices present at the network edge but there exists neither a common fog computing architecture nor a common accepted concept of what is fog node, how and where to deploy the fog node or how it supports real-time Internet of Things (IoT) service execution. Edge devices such as the switch, router, gateway, mobile phones, smart car etc., are the candidates for deployment of fog nodes but the deployment differs according to the application. In this work, we have taken gateways as candidates for fog node deployment. The gateway collects data from smart sensors, but it does not have any pre-processing or decision-making capabilities. Therefore, the gateway is made smarter with Fog capabilities and named as Fog Smart Gateway (FSG). Virtual Machines (VMs) facilitated with distributed Fog nodes take care of the IoT traffic processing. We optimized the number of fog nodes to be deployed for bringing a reduction in the overall latency incurred caused by traffic processing and aggregation. Our results show that the optimal deployment of fog nodes in the IoT setup cause a reduction in latency in comparison to conventional method of IoT data processing in a cloud system.

***Keywords* – Edge devices, Fog computing, Fog node, Service latency, Virtual Machines**

I. INTRODUCTION

Fog computing is a concept that provides services at the network edge and involves smart Gateways named Fog Smart Gateways (FSG). Fog nodes are deployed in the network near the users to handle the services. In this architecture prior processing of data is done locally before it is sent to the cloud. The major issues and challenges of architecture design for edge-centric IoT services are discovering fog nodes, data caching, partitioning. This platform supports micro-service delivery with reduced latency; bandwidth and network load for resource constrained devices along with taking care of service resiliency and localization. It has potential to offer delay sensitive services for applications. It also supports added security, scalability, density of devices and mobility. IoT edge devices increase the computing resources to perform big data analytics. Fog node is a functional and conceptual entity in fog computing. A fog node is a physical element to deploy fog computing. Fog node can provide infrastructure for IoT services' execution. The common characteristics of fog node are that it is distributed and heterogeneous in nature, volatile, highly mobile and supports embedded computing, storage and networking capabilities for easy deployment of IoT applications or services. Most of the authors have not discussed implementation strategy of fog node.

II. RELATED WORKS

A fog network consists of a several fog nodes, and each fog node resides in a base station or an access point such as switch, gateway, router etc. We require a system that has efficient mechanism to choose the edge devices based on functionality and characteristics of fog node to build a fog network. With virtualization technologies, a fog node has the capacity to run multiple virtual machines (VMs) on its own physical machine simultaneously, and a VM can be duplicated into multiple copies and placed in multiple fog nodes [1]. In [2] Xiao et al. put forward an effective way for the design and placement of DCs in optimal manner that focused on improving the QoS in terms of cost efficiency and service latency. However, the process migration within DCs is an overhead degrading the performance for billions of processes in IoT. The smart gateway is proposed as a fog node in [3], [4], the micro data centers option put forward in [5], or the use of fog nodes in Information Centric Networking as cache is given in [6]. Tang et al. [7] presented the three layer fog computing concept for anlasing the big data generated in the smart cities. CISCO edge routers were used for implementation of fog computing in the paper as the concept was initially introduced by Cisco in [8]. An interesting example has been pointed up in [9] that mentions about sharing of computation resources of smart phone only in case the phones are in connection with the grid and it is not enough to handle highly demanding scenarios. Kumar et al. [10] developed mini-clouds which are distributed cloud data centers that facilitate data replication among each other. Masip-Bruin et al. [11] presented a Fog-to-Cloud architecture comprising a layered management structure that integrates different heterogeneous fog layers into hierarchal architecture. A control plane within the Fog-to-Cloud architecture exists that is responsible for distribution of the atomic services among the available edge nodes. The service allocation process aims at reducing the service allocation delay, providing load balance and energy-usage balance among the distinct fogs. Narendra et al. [12] proposed a strategy for optimal placement of mini-cloud in order to minimize latency incurred in data collection from IoT devices; and data migration amongst mini clouds to deal with storage capacity issues along with minimization of access latency . Malandrino et al. [13] present a work where high server utilization is achieved and application latency incurred is low, but the best approach is dependent on the strategy of deployment by the individual network operators and geographic specifications of the cities. Most of the research issues in fog computing [14],[15],[16],[17] the service latency , network traffic and power consumption are reduced by fog computing architecture. We optimized the number of fog nodes to minimize the latency. We consider fog nodes location as a smart IoT gateway and optimally placed fog nodes.

III. FOG COMPUTING ARCHITECTURE

In our architecture (Figure 1), the IoT service network consists of four layers. Huge volumes of data are generated by the IoT devices and the responsibility of aggregating and processing the data lies with the networking elements.

.

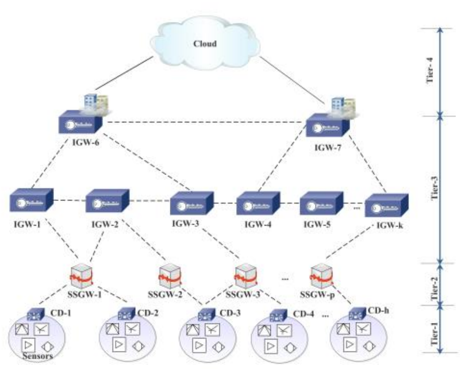

Fig 1: Fog Computing Architecture for IoT Services

(A) *Tier 1:* This is the ground-level layer that includes all the smart sensor nodes (SSNs) with unique IPv6 addresses, and transmit information wirelessly using 6LoWPAN protocol to form a mesh network. SSN is a collection of sensors and actuators. These are responsible for sensing environment data and transmitting to its immediate upper layer. There can be instructions from the upper layer to the actuator to perform an action. A typical smart city scenario has hundreds of networks, pertaining the different domains, deployed all over its geographical area. Each of these networks is coordinated by a Coordinating Device (CD). A CD is known differently in different networks namely Cluster Head(CH) in sensor networks, Access Point(AP) in WiFi networks and Reader in Radio-Frequency Identification(RFID) network etc.

(B) *Tier 2: CD*s need to transmit their data to the Internet for efficient execution of their corresponding applications. This transmission of data is facilitated by the device known as Solution Specific Gateways (*SSGW)* or IoT Gateway (*IGW)*. *CD*s can only communicate through one specific technology and are connected to at least one SSGW/IGW. However, an SSGW is a wireless device which supports technologies of all the *CD*s associated with it. Two SSGW to be connected if and only if they are in each other's range and support at least one mutually common technology, else, they are connected through an *IGW* . *SSGW* s route the data received from *CD*s associated with them to the *IGW* s. The *SSGW* should also ensure the coverage of the *CD*s. Wireless Mesh Network is as close as it can get to the IoT network with one fundamental difference. All gateways in a wireless mesh network support the same set of technologies whereas *SSGW* in IoT supports different sets of technologies. Each *IGW* has a wired connection to the Internet and sends the data received from the *SSGW* s to the upper layer.

(C) *Tier 3:* This tier made up of edge devices for temporary store, process and analyze the received information. FSGs collect data from CDs. Fog Nodes (FNs) are placed within IoT gateways specific to geographic locations. Each FSG serves multiple gateways within its proximity. The FSG is capable of load balancing, service management, resource provisioning of IoT gateways.

(D) *Tier 4:* Each IGW connected to a cloud data center by a wired network. The evolving technology and consolidated computational paradigms have resulted in Cloud Computing.
Virtualization is the prime factor that forms basis of Cloud computing. It brings out from one physical computing device, one or more virtual devices, making each device easily usable and managed to perform tasks.

IV. QoS METRICS

A. *Service Latency*

The service delay is the requested transmission delay and processing delay. We assume that the communication delay between SSNs is considered insignificant. Let δ_{cd_sg} and δ_{sg_igw}, δ_{igw_sfg} be the respective delay incurred in the transmission of a data packet from a CD to its corresponding SSGW, from a SSGW to its corresponding IGW, and from

IGW to a smart fog gateway respectively. $\gamma_{sg}, \gamma_{igw}$ and γ_{sfg} are the processing latency of SSGW, IGW and smart fog gateway for a data packet. Thus, the total transmission latency, Φ_{sfg}, for all the data packets of req_i request running within FN_i is given by

$$\Phi_{sfg} = (\delta_{cd\_sg}\alpha + \delta_{sg\_igw}\beta + \delta_{igw\_sfg}\theta) + (\gamma_{sg}\alpha + \gamma_{igw}\beta + \gamma_{sfg}\theta) \quad (1)$$

where, α, β, and θ $(\alpha > \beta > \theta)$ represents the total number of data packets as sent by CD, SSGW, and IGW respectively.

V. PROBLEM FORMULATION

We placed fog nodes into a gateway from where it can access maximum gateways data and delay is also minimized. Each node transmits data to only one fog node. The IoT network is modeled as a graph *G*(*V*, *E*) where *V* represents the set of nodes (gateways in the network) and *E* denotes the set of undirected edges(link). Propagation latencies are given by the edge weights, where shortest path from node *v is denoted by d*(*v*, *s*), $s \in V$, and the number of nodes $n = |V|$. $S \subset V$ is a set of *k* number of fog nodes which are placed within gateways.

The distance matrix FDM stores the calculated shortest path latency for each node pair such that { FDM_{ij}|i, j ∈ n and FDM_{ii} = 0, FDM_{ij} = FDM_{ji} }. In the worst-case, if there is no limitation of fog nodes required to set up, the solution is to place a fog nodes at each gateway, but for the best case, the number of fog nodes should be restricted to $1 < k < n$. Hence our problem is to minimize the latencies between gateways to fog nodes of the network. Selection of a gateway for fog node is represented by a binary selection variable ψ_j, where *j* = 1,2, ..., *k*.

$$\psi_j = \begin{cases} 1, & \text{if node } v_j \text{ is selected for fog node placement} \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

Let *D*(*S*) represent the total latency between gateways to fog nodes.

$$D(S) = \sum_{j=1}^{f}\sum_{i=1}^{n} \min\left(d\left(v_i, \Psi_j\right)\right) \quad (3)$$

$$D(S') = \min\left(D(S)\right) \quad (4)$$

Given the desired number of fog nodes *k* , there is a finite set of $\binom{n}{k}$ possible placements. The objective is to determine the placement from the all the possible options, such that the overall latency *D*(*S'*) would be minimum.

VI. ALGORITHM FOR FOG NODE PLACEMENT

This section discusses the details of proposed fog node placement algorithm. We are finding the appropriate mapping between gateways and fog nodes. We are applying *k*-means clustering with some modification to solve our problem. The algorithms find the *f* number of fog nodes considering distance as a metric between fog nodes to gateways. Our algorithm discusses the optimum arrangement of fog nodes into the selected gateways. The *k*-means clustering method gives different results for latency using different techniques for choosing the initial centroid. Six different techniques have been used for the selection of initial centroids and the final centroid of each cluster is chosen as the location for fog node placement. The Forgy method selects *f* random values from the *n* locations of the gateways. The Midpoint Method divides the set of gateways into f partitions and takes the midpoint of the partition as the initial centroid. The Split Tree Method determines the centroids by taking mean of different partitions. The entire dataset is the first partition and its mean the first centroid. It is further divided to get the second and third partition and hence the respective centroids. The division of previous partition to get two new partition continues until the number reaches *f*. The Sorted Split Tree Method is similar to Split Tree Method except first the dataset is arranged in sorted order of distance from origin. The Forward Difference Method calculates the difference of adjacent gateways and f centroids are given by mean of locations of gateways giving the *f*-maximum difference values. The Sorted Forward Difference Method is similar to Forward Difference Method except first the dataset is arranged in sorted order of distance from origin. Out of the six techniques Split Tree Method gives the best result.

Algorithm 1: Fog Node Placement

Input: FDM: nxn delay matrix of n number of gateways, f: number of fog nodes where FDM ≠ Φ ∧ 1< f < n
Output: Location of the fog nodes

1: Selection_Initial_Fog_Nodes (FDM, f)
2: **while** not convergence **do**
3: **for** i = 1 to n **do**
4: Compute

$$membership\left(s_j | v_i\right) \ \forall \ membership\left(s_j | v_i\right) \in \{0, 1\}$$

5: // Minimize intra distance
membership $\left(s_j | v_i\right)$ = 1 /* if the delay between gateway v_i and fog node s_j is minimal */
membership $\left(s_j | v_i\right)$ = 0 /* otherwise */
6: **end for**
7: **for** i=1 to n **do**
8: **for** j=1 to f **do**

$$s_j = \frac{\sum_{i=1}^{n} membership\left(s_j | v_i\right) v_i}{\sum_{i=1}^{n} membership\left(s_j | v_i\right)}$$

9: **end for**
10: **end for**
11: **end while**

Algorithm 2: Forgy method for selection of initial fog nodes

Input: Selection_Initial_Fog_Nodes (FDM, f)
Output: Initial locations of fog nodes

1: **for** j=1 to f **do**
2: s_j = Select random v_i , where s_j is the Centroid of Cluster(j) and v_i is the gateway
3: **end for**

Algorithm 3: Mid Point method for selection of initial fog nodes

Input: Selection_Initial_Fog_Nodes (FDM, f)
Output: Initial location of fog nodes

1: partition_size = n/f
2: left = 0
3: right = partition_size
4: **for** j =1 to f **do**
5: s_j = mid-point of [v_i (left), v_i (right)], where s_j is the Centroid of Cluster(j) and v_i is the gateway
6: left = right+1
7: right = right+ partition_size
8: **end for**

Algorithm 4: Split Tree method for selection of initial fog nodes

Input: Selection_Initial_Fog_Nodes (FDM, f)
Output: Initial location of fog nodes

1: i=1
2: partition=set of locations of gateways
3: **for** i=1 to f **do**
4: **if** (i==1) **then**
5: C_i = mean of partition, where C_i is the Centroid of Cluster(i)
6: i=i+1
7: **else**
8: Divide the partition into two halves
9: **for** each new partition **do**
10: **if** (i<=k) **then**
11: C_i = mean of partition
12: i=i+1
13: **end if**
14: **end for**
15: **end if**
16: **end for**

Algorithm 5: Sorted Split Tree method for selection of initial fog nodes

Input: Selection_Initial_Fog_Nodes (FDM, f)
Output: Initial location of fog nodes

1: **for** j=1 to n **do**
2: d_j = Distance of gateways from origin
3: **end for**
4: Sort gateways on basis of distance from origin
5: i=1
6: partition=set of locations of gateways
7: **for** i=1 to f **do**
8: **if** (i==1) **then**
9: C_i = mean of partition, where C_i is the Centroid of Cluster(i)
10: i=i+1
11: **else**
12: Divide the partition into two halves
13: **for** each new partition **do**
14: **if** (i<=k) **then**
15: C_i = mean of partition
16: i=i+1
17: **end if**
18: **end for**
19: **end if**
20: **end for**

Algorithm 6: Forward Difference method for selection of initial fog nodes

Input: Selection_Initial_Fog_Nodes (FDM, f)
Output: Initial location of fog nodes

1: Calculate difference of gateway distance in pairs
2: maxDifference[f]= f maximum differences
3: maxDifferenceIndex[1:f]=corresponding index of the gateways

4: right=0
5: **for** i=1 to f **do**
6: left=right+1
7: right=maxDifferenceIndex(i);

8: C_i=mean of [G(left), G(right)],where C_i is the Centroid of Cluster(i) and G is the gateway
9: **end for**

Algorithm 7: Sorted Forward Difference method for selection of initial fog nodes

---

Input: Selection_Initial_Fog_Nodes (FDM, f)
Output: Initial location of fog nodes

1: **for** j=1 to n **do**
2: d_j = Distance of sensor from origin
3: **end for**
4: Sort sensors on basis of distance from origin
5: Calculate difference of gateway distance in pairs
6: maxDifference[f]= f maximum differences
7: maxDifferenceIndex[1:f] = corresponding index of the gateways
8: right=0
9: **for** i=1 to f **do**
10: left=right+1
11: right=maxDifferenceIndex(i);
12: C_i=mean of [G(left), G(right)],where C_i is the Centroid of Cluster(i) and G is the gateway
13: **end for**

VII. SIMULATION AND RESULTS

We have performed the simulation in the iFogSim simulator and run on the workstation equipped with Intel Core i7, 18 core processor, and 64 GB RAM. IoT gateways are assumed to be randomly distributed. We are fixing the number of gateways to 32 to 512 and varying the number of fog node from 1 to 10. Data transfer from IGW to FSG in the form of the packet and the size of the packet has been ranged between 34-65550 bytes. The instruction size is fixed to 64 bits. Packet arrival is assumed to be Poisson distribution with an average arrival rate of packet for each node set to 1 packet per second. Figure 2 shows the latency (in milliseconds) and the number of fog nodes of the system for the optimal fog nodes placement algorithm. We analyze the service latency and the number of fog nodes. We observed that after placing 6 mini-clouds the latency does not decrease that much. So, we can conclude that minimum 6 mini-clouds required reducing the service latency.

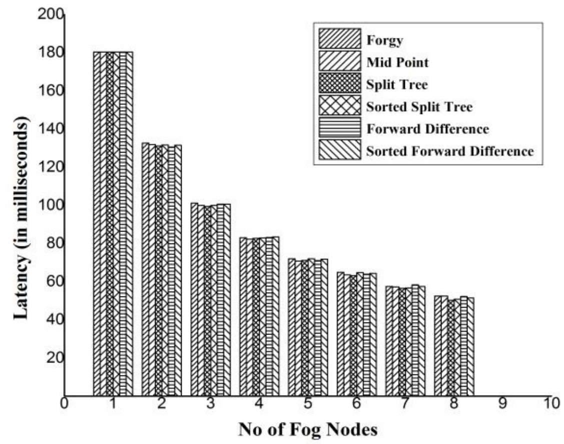

Fig 2 : (a) Latency Vs. No of fog nodes for 32 gateways

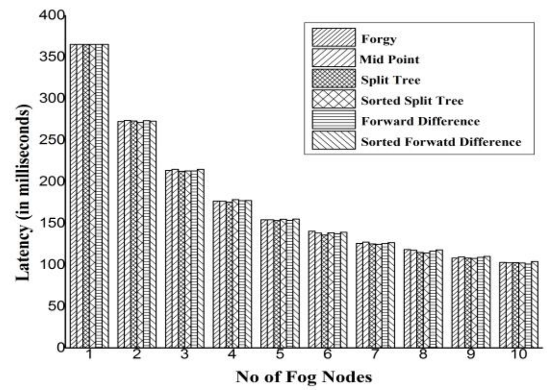

Fig 2: (b) Latency Vs. No of fog nodes for 64 gateways

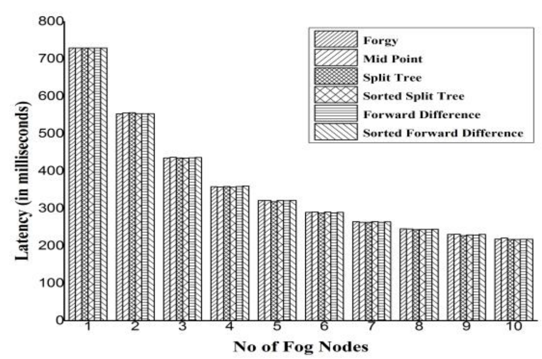

Fig 2: (c) Latency Vs. No of fog nodes for 128 gateways

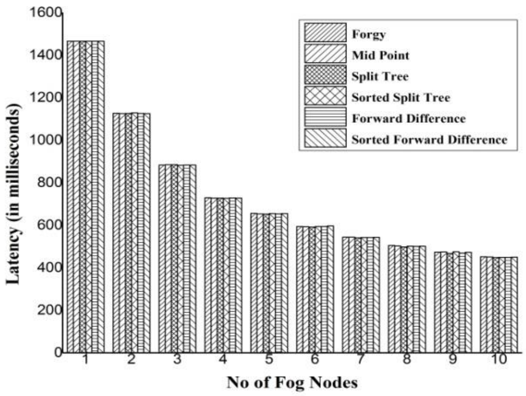

Fig 2 : (d) Latency Vs. No of fog nodes for 256 gateways

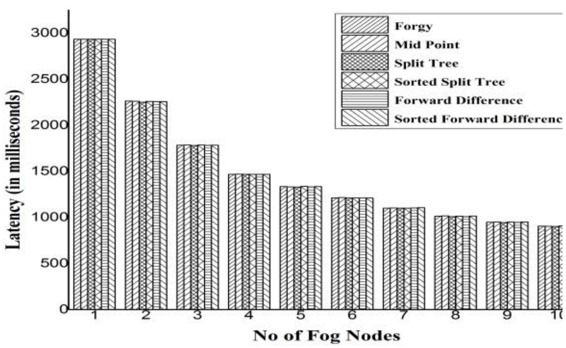

Fig 2 : (e) Latency Vs. No of fog nodes for 512 gateways

IX. CONCLUSION AND FUTURE SCOPE

In this study, we have seen the fog smart gateway placement for IoT services. It was observed that the service latency and power consumption in fog computing environment are significantly lower for a large number of real-time, low latency applications. To add further to the work in future, we plan an extension to the current work as placement of fog nodes and services in a fog computing environment considering load balancing.

ACKNOWLEDGEMENT

This research was supported by Media Lab Asia (Visvesvaraya Ph.D. Scheme for Electronics and IT, Project Code-CSVSE) under the department of MeitY government of India and carried out at Cloud Computing Research Laboratory, Department of CSE, National Institute of Technology Rourkela, India.

REFERENCES

[1] Y. J. Yu, T. C. Chiu, A. C. Pang, M. F. Chen and J. Liu, "Virtual machine placement for backhaul traffic minimization in fog radio access networks," *2017* IEEE International Conference on Communications (ICC), Paris, 2017, pp. 1-7.

[2] J. Xiao, H. Wen, B. Wu, X. Jiang, P.-H. Ho, and L. Zhang, Joint design on dcn placement and survivable cloud service provision over all-optical mesh networks, IEEE Trans. on Communications, vol. 62, no. 1, pp. 235-245, January 2014.

[3] M. Aazam and E. N. Huh, "Fog Computing and Smart Gateway Based Communication for Cloud of Things," 2014 International Conference on Future Internet of Things and Cloud, Barcelona, 2014, pp. 464-470.

[4] T. N. Gia, M. Jiang, A. M. Rahmani, T. Westerlund, P. Liljeberg and H. Tenhunen, "Fog Computing in Healthcare Internet of Things: A Case Study on ECG Feature Extraction," 2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing, Liverpool, 2015, pp. 356-363.

[5] M. Aazam and E. N. Huh, "Fog Computing Micro Datacenter Based Dynamic Resource Estimation and Pricing Model for IoT," 2015 IEEE 29th International Conference on Advanced Information Networking and Applications*, Gwangiu, 2015, pp. 687-694.*

[6] I. Abdullahi, S. Arif, S. Hassan, "Ubiquitous shift with information centric network caching using fog computing" in Computational Intelligence in Information Systems, Cham, Switzerland: Springer, pp. 327-335, 2015.

[7] B. Tang, Z. Chen, G. Hefferman, T. Wei, H. He, Q. Yang, "A hierarchical distributed fog computing architecture for big data analysis in smart cities", ASE BD&SI '15: Proceedings of the ASE BigData & Social Informatics*, pp. 28:1-28:6, 2015.*

[8] Cisco Fog Computing Solutions: Unleash the power of the Internet of Things.

[9] F. Busching, S. Schildt and L. Wolf, "DroidCluster: Towards Smartphone Cluster Computing -- The Streets are Paved with Potential Computer Clusters," 2012 32nd International Conference on Distributed Computing Systems Workshops*, Macau, 2012, pp. 114-117.*

[10] A. Kumar, N. C. Narendra and U. Bellur, "Uploading and Replicating Internet of Things (IoT) Data on Distributed Cloud Storage," 2016 IEEE 9th International Conference on Cloud Computing (CLOUD), San Francisco, CA, 2016, pp. 670-677.

[11] X. Masip-Bruin, E. Marín-Tordera, G. Tashakor, A. Jukan and G. J. Ren, "Foggy clouds and cloudy fogs: a real need for coordinated management of fog-to-cloud computing systems," in IEEE Wireless Communications, vol. 23, no. 5, pp. 120-128, October 2016.

[12] N. C. Narendra, K. Koorapati and V. Ujja, "Towards Cloud-Based Decentralized Storage for Internet of Things Data," 2015 IEEE International Conference on Cloud Computing in Emerging Markets (CCEM), *Bangalore, 2015, pp. 160-168.*

[13] F. Malandrino, S. Kirkpatrick, C.-F. Chiasserini, "How Close to the Edge? Delay/Utilization Trends in MEC", Proceedings of the 2016 ACM Workshop on Cloud-Assisted Networking*, pp. 37-42, 2016.*

[14] S. K. Mishra, D. Putha, J. J. P. C. Rodrigues, B. Sahoo and E. Dutkiewicz, "Sustainable Service Allocation using Metaheuristic Technique in Fog Server for Industrial Applications," in *IEEE Transactions on Industrial Informatics*, vol. PP, no. 99, pp. 1-1.

[15] N. Chen, Y. Chen, Y. You, H. Ling, P. Liang and R. Zimmermann, "Dynamic Urban Surveillance Video Stream Processing Using Fog Computing," *2016* IEEE Second International Conference on Multimedia Big Data (BigMM), Taipei, 2016, pp. 105-112.

[16] A. Brogi and S. Forti, "QoS-Aware Deployment of IoT Applications Through the Fog," in IEEE Internet of Things Journal, vol. 4, no. 5, pp. 1185-1192, Oct. 2017.

[17] S. Sarkar, S. Misra, "Theoretical modelling of fog computing: A green computing paradigm to support IoT applications", *IET Netw.*, vol. 5, no. 2, pp. 23-29, Feb. 2016.