Turkish Online Journal of Qualitative Inquiry (TOJQI)
Volume 12, Issue 8 July, 2021: 5946 – 5955

Research Article

An Approach for Optimal Placement of data in Fog Computing

Govind Murari Upadhyay and Shashi Kant Gupta

School of Engineering and Technology, ITM University, Gwalior, India

Email: itmuphd18CS02@itmuniversity.ac.in, shashikantgupta@itmuniversity.ac.in

Abstract

The widespread usage of the Internet of Things in our daily interactions and the ever-increasing demand for processing capacity improve living quality. To address the needs of various IoT applications, service quality (QoS) is an important aspect of performance evaluation. The computer standard paradigm is wasteful in consideration of latency sensor applications provided by IoT sensors. Fog computing is a promising way to provide cloud services while avoiding the limitations of traditional computers. We offer a way for lowering fog computing electricity usage in this paper so that services can strategically use both active and inactive devices. It is a research paper. Situating a service is undeniably a problem of integrated optimization. The optimally designed solution stretches loads into another fog node with a service that reduces fog node energy. Fog Computing can help us save energy in this way.

Keywords: Fog computing, Cloud computing genetic algorithm.

I. INTRODUCTION

Fog computing is emerging as a promising paradigmto perform distributed, low-latency computation by jointly exploiting end-user devices' radio and computing resourcesand cloud servers.Through the maturing of wireless communication technology and the development of sensor/actuator and radiofrequency identification technology, the Internet of Things (IoT) continues to increase and break new ground in everyday life. IoT Furthermore, the terminals are miniaturized to provide services in various applications based on sensors, netboards, and intelligently made equipment. Sensors are employed in the Internet of Things (IoT) to gather and transport data to a sophisticated cloud computing system that is properly memorized. Because the number of terminals has lately increased, the volume of data transmitted from the terminals to the cloud has increased significantly. This resulted in latency and congestion issues with cloud computing systems. Fog Computing has provided a realistic answer. By expanding local processing and storage, fog node devices can share more of the processing load that was previously carried to the cloud. This decreases network traffic and delays while also filling data storage and transmission gaps.

Fog Placement of Computer Services Fog Computing is a hazy and highly scalable entity that provides cloud users with a diverse set of services and charges, as well as dynamic content and on-demand services. Many customers refer to a service application method for putting service on a fog-computer system without breaking the SLA and jeopardizing QoS over the Internet. To maximize fog computing, we require efficient placement services that can meet various goals [2],[3]. I Energy use should be kept to a minimum. ii) Lower the cost of communication. Communication is expensive. iv) The load balance should be as light as possible. iii)Decrease the layer's thickness. For request analysis and services submitted on IoT devices, the requirements of your fog cluster and the data accessible to this cluster must be monitored and known by each Fog Node Manager and Cloud Fog controller. The scheduler[4] can carry out a service placement plan and dispatch service requests to a certain fog resource. Only if cloud control necessitates more cloud resources will each Fog cluster be used. Every

fog cluster is thought to be self-contained. We want to optimize cost-energy-time resources in this scenario by shifting our services to meet the QoS requirements[5]. Only by maximizing the computing capabilities of fog nodes and data sources and selecting the appropriate fog cluster can this be accomplished. If one nebulizer is overpowered, another nebulizer cluster should be used.

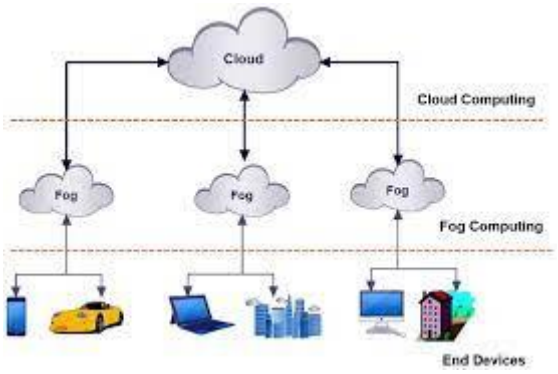

Figure 1: Basic fog computing model[2]

If the neighbourhood cluster[6] fails to fulfil the necessary QoS, for example owing to a shortage of deadlines, cloud services are sought. We will be too late if non-fog protected services[7] are restricted[7]. To respond to the above-mentioned difficulty of placing services as a Fog Computing (SPFC) question, a question must be expressed. The fog node manager, which receives and distributes IoT applications into fog nodes or fog clusters, solves this problem. The rest of the paper is organised as follows: Section 2 covers IoT and fog computers; Section 3 covers terminal layer design and the fog-layer allocation model; and Section 4 covers experimental results and design model analysis information. Section 5 of the second section covers the findings and future study.

II. RELATED WORK

Several longitudinal studies in fog computing, including IoT and service placement with parameters, delay, and QoS, have maximized resource utilization.

P. Maiti, et al.[7] address the challenge of service placement; fog computing consideredseveral criteria. Various scholars have offered various strategies; however, performance analysis is critical to implementing these techniques. Interest in the energy efficiency and quality of service of an Internet-based virtual computing environment that tries to provide fog services utilizing a dynamic mix of data centres and other multi-scale internet computer resources has grown in recent years.

Al-Ammar and Ben-Ammar [8] To overcome the problem of fog computing, several scientists use approaches such as genetic algorithms, heuristics, greedy heuristics, and metaheuristics. However, no attempt was made to investigate the relationship between latency and energy consumption in the fog computing setting.

V. Yadav et. al. [9]A vast body of fog computing literature has been published, encompassing a wide range of topics. However, to compute energy consumption and quantify energy, the study ignores other factors like clocks, fan velocity, and so on. A number of researchers has reported the following. Cloud computing has made a significant contribution to the information era.

The Holy Spirit et. al. [10].Data processing has proven to be a strong tool thanks to cloud computing. On the other hand, cloud computing has several intrinsic flaws, such as high cost, scalability, long latency, bandwidth limitations, mobility support, location awareness, and reliability. Cloud computing as a centralised system can execute a huge number of calculation processes and services.

J. Gideon et al. [11]However, network connectivity could be a significant cloud computing concern for big amounts of data. As a result, there is a significant lag (i.e., time for data to travel from one point to another). The number of devices expanded, particularly for applications that required a quick response time.

Rajawat et al.[12], Anand S. is a writer who lives in India. Cloud computing faces significant mobility and location sensitivity difficulties due to the distributed nature of IoT devices. Fog computing was created to address the majority of cloud-related issues. IT equipment is close by and can help with various issues, such as high latency, communication[13], storage, control, service, and processing. Fog nodes are scattered around the network, assisting in resolving mobility issues, scalability, and smooth bandwidth interruptions. Fog computing has several advantages, including bandwidth savings, mobility support, low latency, heterogeneity, geographic spread, and low power consumption.

III. PROPOSED METHODOLOGY

Terminals, household appliances, wireless sensors, and actuators are among the IoT devices used in diverse locations. These systems send data at a higher level to process and execute it. Sensors in the Internet of Things monitor and submit applications for processing and filtering gates, while actuators supply service data.

The nebula is a nebula within a nebula. The fog is separated into two levels in the central layer: Terminals, mobile phones, wearables, and tablets are examples of fog cells[21]. 2. Knots in the fog 2. Inventive+ phrasing (e.g., routers, switches, access points). In IoT devices with sensors and actuators, fog cells represent virtual resources. These virtual resources are used to deploy and run arbitrary Internet of Things (IoT) services. Fog cells are virtual resources at the bottom of the fog layer and require less treatment and storage than fog equipment. Sensor and control services are set up and operated using these virtual resources. Fog nodes are fog cells that serve as points of entry and processing services. Fog nodes with multiple levels are possible. On each level, several fog nodes are split into clusters. FON selects which services are to be supplied through fog nodes in each cluster using the service positioning technique. Two fog nodes belong to the same cluster if the following criteria are met: The hierarchical structure of the parent nodes is the same, and (ii) the link delay is below the threshold.

The three-tier architecture's applications and services

To help you understand how to use the suggested 3-tier architecture, we'll use certain terminology related to IoT applications and services. The first layer has been created. An Internet of Things (IoT) application is a particular application that leverages the Internet to provide information, action features, or information to customers who have requested it. A request is a set of services (or tasks) that are described in the following way. The second layer is A service is the tiniest piece of software that performs a certain task. Sensing, actuating, processing, and storing are the four types of tasks that can be classified. The number of nodes/devices to be employed depends on the task (e.g., a storing service cannot be conducted at a lightweight sensor without storage capacity). In this article, the terms service and 'task' are used interchangeably. The third layer is Any device that performs three-tier architectural work in response to consumer requests. For example, IoT (Things) devices can provide cloud or operational services, whereas cloud-based devices can give IoT, Fog, DC, or cloud-based fog or DC services and provide storage. The user application, which submits requests, is the fourth level. We presume you already have the code for IoT apps and services in your 3-tier connected devices (providers). Which apps or services are estimated is unknown to the architecture. It only knows what kinds of suppliers are required to meet customer expectations. The provider overlays topology and the required resources for each service while keeping track of its time to respond to sensor action. Assume that SCEs are utilized in the pre-definition and documentation of applications and services, with consumers having the ability to request specific applications. When a customer fills out a form, a procedure is created. By sharing identical processing methods and functionalities, CC fog computing enables it to supply expandable non-trivalent computer services beyond the core network boundary (virtualization, multi-denomination[14], etc.). Fog computing is more relevant because of the following transformation features: Latency applications in the environment, (iii) massive distributed control systems (e.g., intelligent grid, connected rails, STLS), and (iv) geologically transmitted apps all irritate me (e.g. sensor network for the environment). Fog computing of processing, networking,

and storage services performs dynamic metamorphosis into fog nodes, cloud, and IoT[17]. Fug must contact other fogs, goods, or people for the cloud to function. Instead, the interfaces should allow for dynamic transfers between calculation, storage, and control entities. Using Quality of Service for Fog End-user satisfaction was analyzed and effectively regulated using calculations from a central location. There is a cloud fog.The clouds are actually fog. Fog. Nebulism, cloud, nebula, cloud, or nebular coordination may be required for back-to-back services. Fog and cloud to better management; fog to process and compare data and other necessary aspects; fog and cloud to distribute and plan on-demand fog nodes; and fog and cloud to improve fog node processing. It is necessary to define the data and services that will be carried via fog and cloud. Fog or cloud behaviour should be influenced by the frequency and granularity of data and information. To enable fog-to-fog processing, fog nodes should contain a pool of resources. With the notion of priority node functionality, all[17] fog nodes could share storage, computation, and processing capabilities for one or more applications. As a backup, many fog nodes could work together. Things To deliver services for heterogeneous IoT device topologies, Fog Computing Internet employs a differential authentication identity technique (e.g., smart devices and sensors). User-friendly, resource-friendly, and secure IoT connectivity to fog services are the most important requirements for IoT access to Fog-to-User interfaces. [18]. The particle's beginning position is determined via a random variation of the greedy algorithm. The velocity of convergence, not the population, determines the PSO's performance. The population was predicted to be 25 particles based on these statistics. These particles' velocity is determined by the same cognitive and social values. Initially, the m inertia parameter was considered to be constant.

$$m = m_{\max} - \frac{m_{\max} - m_{\min}}{\theta_{\max}} \times \theta$$

On the other hand, test results demonstrate that the number of tests should be high at first, then gradually lowered as the answers get more refined. In m, the following linear declining function might be followed: mmax and mmin are the maximum and lowest inertia weights in our simulations, set at 0.9 and 0.4, respectively. We limit the number of iterations to 250 to keep the algorithm running as quickly as feasible.

With maximal repeats, several studies with varied numbers of particles were undertaken. Under the conditions we investigated, increasing the number of iterations or partial elements did not improve the performance of our technique.

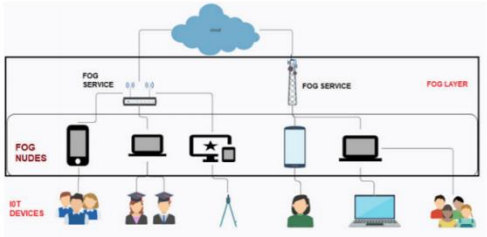

Figure 2: Proposed approach Diagram

A fog node network that extended multiple types of heterogeneous resources from the user layer to the network's edge cloud was previously classified as fog computing. Because the location of these resources is so important, it's considered a challenge in the fog computing environment. The Fog application's structure ensures that IoT apps are used correctly and efficiently. When it comes to fog computing, Fog node applications are NP-hard because of the placement challenges discussed earlier. A procedure was proposed for achieving the greatest and most effective results. It was suggested. Placement of fog computer applications and response time and expenses To reduce the data size, the particle swarm approach was applied. Sensor nodes in a network collect data from their surroundings. When the data is received, the nodes are transmitted to the gates, forwarded to the fog systems. Data will be saved and analyzed in the fog before being transferred to the cloud. Each application contains several modules.

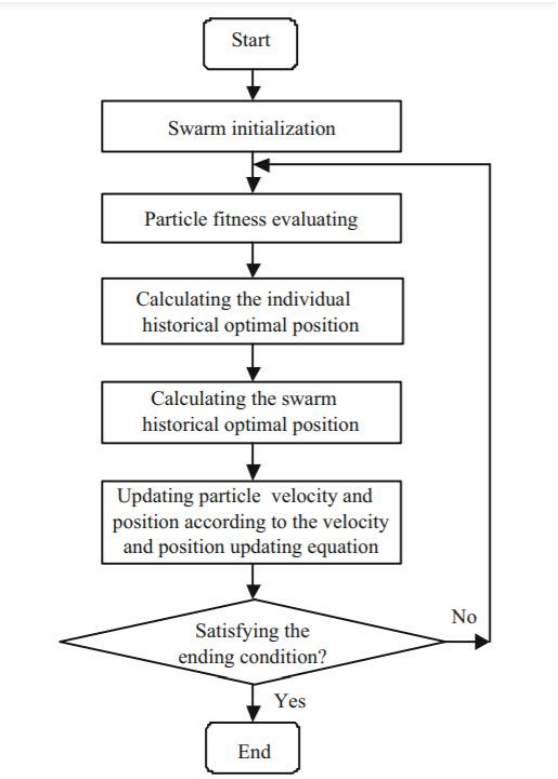

Figure 3: Optimal Placement Of Data On Fog Computing Environment using Swam optimization algorithm

PSO is a population-based swarm optimization system that uses a population-based algorithm based on swarm fodder behaviour. Every point in a SPA holds the position of the best performance in a given neighbourhood, and it uses this information to update its position using the equations (coefficient of restriction):: Each point in the SPA has the following information: In the Wellness Center:

Alternatively (weight of inertia):

The user can specify the values for l, 1, 2, and the maximum permitted speed magnitude (normalized with respect ot the bounds). The user can select where the velocity updating rule differs in the R1 and R2 definitions in one of five variants:

$$\mathbf{v}_{i+1} = \omega\left(\mathbf{v}_i + \eta_1\mathbf{r}_1 \cdot \left(\mathbf{x}_i - \mathbf{x}_i^l\right) + \eta_2\mathbf{r}_2 \cdot \left(\mathbf{x}_i - \mathbf{x}^g\right)\right)$$
$$\mathbf{x}_{i+1} = \mathbf{x}_i + \mathbf{v}_i$$

Many modules are required to run programs on fog devices, which necessitates using a virtual machine.

On the fog gadget, there is a machine. To best distribute virtual machines to the modules[9][10], optimal techniques are utilized.

$$\mathbf{v}_{i+1} = \omega\mathbf{v}_i + \eta_1\mathbf{r}_1 \cdot \left(\mathbf{x}_i - \mathbf{x}_i^l\right) + \eta_2\mathbf{r}_2 \cdot \left(\mathbf{x}_i - \mathbf{x}^g\right)$$
$$\mathbf{x}_{i+1} = \mathbf{x}_i + \mathbf{v}_i$$

- Variant 1: $\mathbf{r}_1 = [r_{11}, r_{12}, \ldots, r_{1n}], \mathbf{r}_2 = [r_{21}, r_{21}, \ldots, r_{2n}]$ ... (inertia weight)
- Variant 2: $\mathbf{r}_1 = [r_{11}, r_{12}, \ldots, r_{1n}], \mathbf{r}_2 = [r_{11}, r_{11}, \ldots, r_{1n}]$ ... (inertia weight)
- Variant 3: $\mathbf{r}_1 = [r_1, r_1, \ldots, r_1], \mathbf{r}_2 = [r_2, r_2, \ldots, r_2]$ ... (inertia weight)
- Variant 4: $\mathbf{r}_1 = [r_1, r_1, \ldots, r_1], \mathbf{r}_2 = [r_1, r_1, \ldots, r_1]$ ... (inertia weight)
- Variant 5: $\mathbf{r}_1 = [r_{11}, r_{12}, \ldots, r_{1n}], \mathbf{r}_2 = [r_{21}, r_{21}, \ldots, r_{2n}]$ ... (constriction coefficient)
- Variant 6: Fully Informed Particle Swarm (FIPS)

1. As previously stated, task planning is an NP-hard problem in the cloud computing context. When there are a lot of players, finding the best option might be fairly tough. Various sophisticated ways of optimization are the standard strategy for reaching an optimum result. One of these algorithms is to use genetic algorithms to provide the approximate best solution[19] and determine the optimal solution of the IOT-FCM model. The genetic algorithm is changed by using a single fitness function derived from multiple training functions and crossing three generations of children. This is the most recent version of the genetic algorithm:
2. Proposed algorithm The Agnihotra is a Hindu deity. The genetic algorithm procedure contains the following elements[1]:
3. Step 1: Determine the number of chromosomes, their generation, mutation rate, and crossover rate.
4. The rate's worth
5. Step 2: Generate chromosome-chromosome numbers for the population and start-up
6. The worth of genes chromosome-chromosome random value
7. Step 3: Repeat steps 4–7 until all generations have been met.
8. Step 4: Determine objective function to assess chromosomal fitness.
9. Step 5: Chromosome selection
10. Step 6: Take a cruise
11. Step 7: Modify
12. Step 8: Compromise (Best Chromosomes)

1. The first step is to initialize the population. To begin, the population is defined, and essential parameters such as population size (P), probability of crossover (pc), and probability of mutation (pm) are determined (pm). The multi-fitness system is linked to the multi-target parameters of the genetic algorithm. The fitness function is an equation for each vector specified as a chromosome in our evolutionary formulation (4).

2. Adaptation The second phase is to select two people from the population as parents who will produce two offspring to achieve improved fitness. The third kid is included in this paper to increase population variety by accumulating parent gene values and generating a new child to find the ideal solution.

3. Variation There are several types of mutations, including Gaussian, consistent, and uniform mutations, among others. Only one gene's value is altered in these mutations, allowing it to boost its chromosomal fitness. The overall influence on the chromosome is minimal, especially if the population is large or the solution is stable[26]. The transformation method has been changed into a multiple gene mutation. We construct multi-mutation chromosomes in order to substitute chromosomes for the population's lowest fitness. As a result, the effect on optimum values was minimised, while the search range was greatly broadened and premature convergence to an optimal, local solution was reduced. The primary goal of mutation is to generate new parent genes.

4. The new chromosomal population established through crossover and mutation processes is merged in this step. Following that, the best individuals with the highest F(C) value are picked as the population for the following generation.

5. To complete the simulation, repeat steps 2–4.

6. The Mapping algorithm employs a contemporaneous method. Application

7. Requests are mapped to Fog equipment in the majority of cases, but not always.

8. the capacity of their application and the criteria for it[6]. So, if you're one of them.

9. The CPU's capability is insufficient to support the Fog device chosen.

10. Application requirements, followed by mapping to form a processing line

11. Fog's node, number eleven.

12. A parameter specifies a list of probable pathways.

13. a method for implementing a leaf-to-root applicationtraversal. The Mapping pseudo-code is provided in Algorithm 3.

Algorithm 3 – Mapping

1: while p ϵ PATHS do ▷ Across all paths

2: placeList := {} ▷ device list

3: while Fog device d ϵ p do ▷ way

4: while module w ϵ app do

5: if all predec. of w are placed then

6: add w to placeList

7: end if

8: end while

9: while module θ ϵ placeList do

10: if θ place on d ϵ p then

11: d := Device

12: Place θ on device d

13: end if

14: end while

15: end while

16: end while

We provide the findings of a fog computing layer delay (power consumption), distance sum, and overall energy consumption simulation experiment. After that, the outcomes of our GA-optimized IoT-FCM model were compared to typical fog algorithms[20]. Fog execution machines use a lot of energy. In order to provide the optimum answer for work planning difficulties, the makepan is typically the major parameter in the traditional max-min technique. In order to produce the optimal answer, the nebulous maximum approach employed in this paper evaluates a variety of criteria

(including delay, distance, and energy consumption). Consumption of energy[21] From the perspective of the IoT end layer, the Routing Protocol should meet the goal of reducing node energy usage. We employed a lot of Sink nodes in the original LIBP[22].to perform the simulation using python language with an anaconda environment. Our proposed approach is very effective to exist one.in figure 2 represent the optimal placement of data in a fog computing environment.

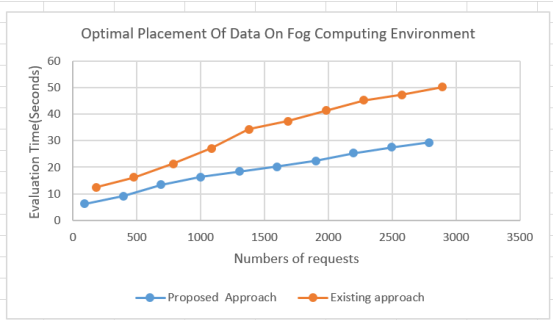

Figure 2:Optimal Placement Of Data On Fog Computing Environment

IoT updated FCM[23] to indicate energy consumption if the LIBP (with multiple Sink Nodes) is used. Each of the sink-knots consumes the greatest energy for each network, at 5.84 percent, 5.71 percent, 6.00 percent, and 5.86 percent, respectively[24]. These statistics are lower than the 6.60 percent[25] energy consumption of the original Sink Node. When multiple[26] sink nodes are employed in each network, the average energy consumption is 4.23 percent, 3.92 percent, 4.38 percent, and 5.07 percent. The average energy usage of a single sink node in the original LIBP was 6.60 percent. When comparing the results, the LIMP protocol utilised by IoT-FCM utilises less energy than the original LIBP protocol. The battery life of the sink nodes will be improved as a result of this.

Conclusion and future work

This document covers the Internet for fog calculations for objects and describes and defines an IoT fog architecture. The IoT-FCM paradigm has two components. A modified genetic algorithm was used to match and assign work to nodes in the fog computing layer, taking into account timing, the distance between fog nodes and users, and fog node power consumption. Simulations were created to demonstrate the strategy's utility. The outcomes were compared to those of a fog-based max-min algorithm and a traditional maximum-minute technique. The IoT-FCM reduces user proximity to the fog node by 38%, compared to traditional maximums of 55% for fog-focused max-minute applications. When compared to other algorithms, IoT-FCM saved an average of 150 KWh of energy. The LIBP protocol was converted to a terminal layer with various sinks as part of the IoTconcept. FCM's The modified LIBP with numerous sink nodes was stronger, more node failure tolerant, and more energetic, according to Cooja Contiki statistics. Given the various environment in which IoT-FCM layers have been simulated, tasks/data were not promptly placed on the terminal layer during the test carried out by the authors of this work. This could, however, be considered in future attempts.

Reference

[1]. Muhammad Rizwan Anawar, Shangguang Wang, Muhammad Azam Zia, Ahmer Khan Jadoon, Umair Akram, Salman Raza, "Fog Computing: An Overview of Big IoT Data Analytics", Wireless Communications and Mobile Computing, vol. 2018, Article ID 7157192, 22 pages, 2018. https://doi.org/10.1155/2018/7157192.

[2]. Atlam, H.F.; Walters, R.J.; Wills, G.B. Fog Computing and the Internet of Things: A Review. *Big Data Cogn. Comput.* **2018**, *2*, 10. https://doi.org/10.3390/bdcc2020010

[3]. H. K. Apat, B. sahoo, P. Maiti and P. Patel, "Review on QoS Aware Resource Management in Fog Computing Environment," 2020 IEEE International Symposium on Sustainable Energy, Signal Processing and Cyber Security (iSSSC), 2020, pp. 1-6, doi: 10.1109/iSSSC50941.2020.9358897.

[4]. A. Mseddi, W. Jaafar, H. Elbiaze and W. Ajib, "Joint Container Placement and Task Provisioning in Dynamic Fog Computing," in IEEE Internet of Things Journal, vol. 6, no. 6, pp. 10028-10040, Dec. 2019, doi: 10.1109/JIOT.2019.2935056.
[5]. Z. Rezazadeh, M. Rezaei and M. Nickray, "LAMP: A Hybrid Fog-Cloud Latency-Aware Module Placement Algorithm for IoT Applications," 2019 5th Conference on Knowledge Based Engineering and Innovation (KBEI), 2019, pp. 845-850, doi: 10.1109/KBEI.2019.8734958.
[6]. T. Hiessl, V. Karagiannis, C. Hochreiner, S. Schulte and M. Nardelli, "Optimal Placement of Stream Processing Operators in the Fog," 2019 IEEE 3rd International Conference on Fog and Edge Computing (ICFEC), 2019, pp. 1-10, doi: 10.1109/CFEC.2019.8733147.
[7]. P. Maiti, J. Shukla, B. Sahoo and A. K. Turuk, "Efficient Data Collection for IoT Services in Edge Computing Environment," 2017 International Conference on Information Technology (ICIT), 2017, pp. 101-106, doi: 10.1109/ICIT.2017.40.
[8]. H. Ben-Ammar and Y. Ghamri-Doudane, "An ICN-based Approach for Service Caching in Edge/Fog Environments," GLOBECOM 2020 - 2020 IEEE Global Communications Conference, 2020, pp. 1-6, doi: 10.1109/GLOBECOM42002.2020.9322305.
[9]. V. Yadav, B. V. Natesha and R. M. R. Guddeti, "GA-PSO: Service Allocation in Fog Computing Environment Using Hybrid Bio-Inspired Algorithm," TENCON 2019 - 2019 IEEE Region 10 Conference (TENCON), 2019, pp. 1280-1285, doi: 10.1109/TENCON.2019.8929234.
[10]. T. S. Nikoui, A. Balador, A. M. Rahmani and Z. Bakhshi, "Cost-Aware Task Scheduling in Fog-Cloud Environment," 2020 CSI/CPSSI International Symposium on Real-Time and Embedded Systems and Technologies (RTEST), 2020, pp. 1-8, doi: 10.1109/RTEST49666.2020.9140118.
[11]. J. Gedeon, M. Stein, L. Wang and M. Muehlhaeuser, "On Scalable In-Network Operator Placement for Edge Computing," 2018 27th International Conference on Computer Communication and Networks (ICCCN), 2018, pp. 1-9, doi: 10.1109/ICCCN.2018.8487419.
[12]. Anand Singh Rajawat Sumit Jain KanishkBarhanpurkar,(2021) Fusion protocol for improving coverage and connectivity WSNs, IET Wireless Sensor Systems, https://doi.org/10.1049/wss2.12018
[13]. Rajawat A.S., Upadhyay P., Upadhyay A. (2021) Novel Deep Learning Model for Uncertainty Prediction in Mobile Computing. In: Arai K., Kapoor S., Bhatia R. (eds) Intelligent Systems and Applications. IntelliSys 2020. Advances in Intelligent Systems and Computing, vol 1250. Springer, Cham. https://doi.org/10.1007/978-3-030-55180-3_49
[14]. Gasmi, K., Dilek, S., Tosun, S. et al. A survey on computation offloading and service placement in fog computing-based IoT. J Supercomput (2021). https://doi.org/10.1007/s11227-021-03941-y
[15]. Ghobaei-Arani, M., Souri, A. & Rahmanian, A.A. Resource Management Approaches in Fog Computing: a Comprehensive Review. J Grid Computing 18, 1–42 (2020). https://doi.org/10.1007/s10723-019-09491-1
[16]. Al-Tarawneh, M.A.B. Bi-objective optimization of application placement in fog computing environments. J Ambient Intell Human Comput (2021). https://doi.org/10.1007/s12652-021-02910-w
[17]. Lin, CC., Deng, DJ., Suwatcharachaitiwong, S. et al. Dynamic Weighted Fog Computing Device Placement Using a Bat-Inspired Algorithm with Dynamic Local Search Selection. Mobile NetwAppl 25, 1805–1815 (2020). https://doi.org/10.1007/s11036-020-01565-9
[18]. Luthra, M., Koldehofe, B. & Steinmetz, R. Transitions for Increased Flexibility in Fog Computing: A Case Study on Complex Event Processing. Informatik Spektrum 42, 244–255 (2019). https://doi.org/10.1007/s00287-019-01191-0
[19]. Huang, T., Lin, W., Li, Y. et al. A Latency-Aware Multiple Data Replicas Placement Strategy for Fog Computing. J Sign Process Syst 91, 1191–1204 (2019). https://doi.org/10.1007/s11265-019-1444-5

[20]. Baranwal, G., Yadav, R. & Vidyarthi, D.P. QoE Aware IoT Application Placement in Fog Computing Using Modified-TOPSIS. *Mobile NetwAppl* **25,** 1816–1832 (2020). https://doi.org/10.1007/s11036-020-01563-x

[21]. Hedhli, A., Mezni, H. A Survey of Service Placement in Cloud Environments. *J Grid Computing* **19,** 23 (2021). https://doi.org/10.1007/s10723-021-09565-z

[22]. Skarlat, O., Nardelli, M., Schulte, S. et al. Optimized IoT service placement in the fog. SOCA 11, 427–443 (2017). https://doi.org/10.1007/s11761-017-0219-8

[23]. Rajawat A.S., Barhanpurkar K., Shaw R.N., Ghosh A. (2021) Risk Detection in Wireless Body Sensor Networks for Health Monitoring Using Hybrid Deep Learning. In: Mekhilef S., Favorskaya M., Pandey R.K., Shaw R.N. (eds) Innovations in Electrical and Electronic Engineering. Lecture Notes in Electrical Engineering, vol 756. Springer, Singapore. https://doi.org/10.1007/978-981-16-0749-3_54

[24]. Kaur, M., Aron, R. A systematic study of load balancing approaches in the fog computing environment. J Supercomput (2021). https://doi.org/10.1007/s11227-020-03600-8

[25]. Bhatia, M., Sood, S.K. & Kaur, S. Quantumized approach of load scheduling in fog computing environment for IoT applications. Computing 102, 1097–1115 (2020). https://doi.org/10.1007/s00607-019-00786-5

[26]. Guerrero, C., Lera, I. & Juiz, C. A lightweight decentralized service placement policy for performance optimization in fog computing. *J Ambient Intell Human Comput* **10,** 2435–2452 (2019). https://doi.org/10.1007/s12652-018-0914-0

[27]. Anand Singh Rajawat Sumit Jain KanishkBarhanpurkar,(2021) Fusion protocol for improving coverage and connectivity WSNs, IET Wireless Sensor Systems, https://doi.org/10.1049/wss2.12018

[28]. Baranwal, G., Vidyarthi, D.P. FONS: a fog orchestrator node selection model to improve application placement in fog computing. J Supercomput (2021). https://doi.org/10.1007/s11227-021-03702-x

[29]. Javadzadeh, G., Rahmani, A.M. Fog Computing Applications in Smart Cities: A Systematic Survey. Wireless Netw 26, 1433–1457 (2020). https://doi.org/10.1007/s11276-019-02208-y

[30]. Martin, J.P., Kandasamy, A. & Chandrasekaran, K. Mobility aware autonomic approach for the migration of application modules in fog computing environment. J Ambient Intell Human Comput 11, 5259–5278 (2020). https://doi.org/10.1007/s12652-020-01854-x

[31]. DadashiGavaber, M., Rajabzadeh, A. MFP: an approach to delay and energy-efficient module placement in IoT applications based on multi-fog. J Ambient Intell Human Comput 12, 7965–7981 (2021). https://doi.org/10.1007/s12652-020-02525-7

[32]. Subbaraj, S., Thiyagarajan, R. &Rengaraj, M. A smart fog computing based real-time secure resource allocation and scheduling strategy using multi-objective crow search algorithm. J Ambient Intell Human Comput (2021). https://doi.org/10.1007/s12652-021-03354-y

[33]. Ramzanpoor, Y., Hosseini Shirvani, M. &Golsorkhtabaramiri, M. Multi-objective fault-tolerant optimization algorithm for deployment of IoT applications on fog computing infrastructure. Complex Intell. Syst. (2021). https://doi.org/10.1007/s40747-021-00368-z

[34]. Madhura, R., Elizabeth, B.L. &Uthariaraj, V.R. An improved list-based task scheduling algorithm for fog computing environment. *Computing* **103,** 1353–1389 (2021). https://doi.org/10.1007/s00607-021-00935-9