JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

TASK SCHEDULING IN FOG ENVIRONMENT USING MACHINE LEARNING

[1]Maitree Desai, [2]Reshma Dayma

[1]Student, [2]Assistant Professor

[1]Department of Computer Engineering,

[1]L. D. College of Engineering, Ahmedabad, Gujarat, India

Abstract

Lots of new smart devices connected to the Internet of Things(IoT) are changing the way we live. In the current cloud computing model, latency, data volume, and bandwidth are all challenges that need to be addressed. Cloud and Internet of Things (IoT) architectures that bring services closer to the edge device are called "fog computing," and they work in conjunction with them. Fog computing's resource management is one of the open issues. The fog computing machine learning application has grown into powerful end-user and high-layer services that can do more analysis and give better answers to tasks. We suggested a Scheduling algorithm based on the ML technique. It uses K-mean clustering for grouping jobs based on time and cost. Clustering makes it easy to discover a collection of jobs per VM with minimal cost. iFogsim is being used to model the proposed approach. The simulation result reveals that our proposed scheduling technique improved execution cost and provided low network usage and better execution time than existing scheduling methods in iFogsim.

Keywords: Fog Computing, Machine Learning, Task Scheduling, K-mean clustering, Internet of Things (IoT).

1. Introduction

Cloud computing has a large presence in the technology industry since it provides end users with a variety of helpful services. Cloud computing offers several benefits, including immense processing power, ample storage, a massive network connecting processing nodes and data sources, and a pay-per-use

approach. Cloud computing is a strong technology that provides these paradigms as well as many other benefits such as flexibility, cheaper costs, scalability, and ease of software installation.

However, despite these benefits, Cloud computing has certain disadvantages. Some of the disadvantages include: the client and Cloud layer may be geographically separated, which can cause transmission delays; there may be a scarcity of resources for task execution; many resources may be idle even if tasks must be performed instantly; and so on. Virtualized Fog computing technology is used to solve these issues[1].

Fog is a layer that sits between end users and cloud data centers. Fog computing can be useful for executing applications that require low latency and real-time responses, depending on the location of the data producer. This layer can include a large number of virtual servers to handle incoming requests.

"Resource allocation is the systematic approach of allocating available resources to the needed Cloud clients over the Internet," according to Agarwal, Yadav, and Yadav [2]. The timing and order in which resources are allotted is critical for maximizing the benefits of employing a virtual server, since the system's throughput may be increased while customers are not overcharged. The availability of resources should ensure that high-priority jobs do not wind up at the bottom of the task queue. This might result in inefficient utilization of virtual servers and possibly company loss. As a result, allocating resources in a prioritized manner to maximize profit is critical and a promising study topic.

Furthermore, machine learning (ML), an important field, has made significant advances in a variety of academic areas, including robotics, neuromorphic computing, computer graphics, natural language processing (NLP), decision-making, and speech recognition. Several researches have been presented to look at ways to use machine learning to solve fog computing issues. In recent years, there has been an increase in the use of machine learning (ML) to improve fog computing applications and deliver fog services, such as efficient resource management, security, latency and energy reduction, and traffic modeling.There are many different types of fog computing devices, sensors, and objects, and each one generates a large amount of data that must be processed. Real-time processing has the potential to improve efficiency. In some cases, it may be necessary. Sensors, devices, and by sending requests, objects will completely utilize resources [42]. As a result, fog computing requires resource management and should be implemented with caution [43]. In this section, we looked at studies that used ml algorithms to manage fog computing resources.

This paper proposes a Scheduling Algorithm which is used to schedule tasks at fog level. A task is scheduled to the VM that plays a role in execution of request/response model in fog computing. We use a k-means clustering algorithm for scheduling fog devices.The default resource scheduler in the simulator equally divides fog device’s resources among all active application modules. Clustering makes it easy to

find a set of tasks for Vm with minimum cost. Therefore, the integration of ML method i.e. Clustering in scheduling tasks in fog computing will give better quality of services(QoS) with low execution cost and low network usage. our key contribution in this work are following:

1. Presentation of Clustering Scheduling in fog computing.
2. Implementation of proposed algorithm in iFogsim.
3. Reduction of Execution Cost.

The remaining paper is organized in the following order, chapter 2 focuses on Literature survey of related work to the research topic. Chapter 3 describes the Research gap. Chapter 4 describes the fog architecture and the design of the system. In chapter 5 the Proposed algorithm of scheduling is discussed. Chapter 6 describes the results from the simulation and comparison. And lastly, the paper concludes with a summary in chapter 7.

2. Literature Survey

There are a variety of scheduling algorithms in distributed systems and the main purpose of scheduling algorithms is to obtain high computational performance and better system execution. The traditional task scheduling algorithms are not compatible to provide scheduling in a fog environment. Scheduling is a process that consider several parameters simultaneously to optimize the execution of requests in fog networks, such as task queue, duration of request, speed of VM, priority of queue. [2]

In [2], the Ant colony algorithm is used to schedule the grouped taska to virtual machines. There are three major steps in the algorithm to group tasks based on time and cost and prioritized tasks then optimal VM is selected with the help of Ant colony algorithm. This algorithm is simulated in iFogsim and improves simulation results by low energy consumption.

In [3], the authors present a novel scheduling which uses knapsack optimization with symbiotic organism search (SOS). They also present a novel application for elderely human's activity detection(EAHD) in smart homes for detection of health issues of aged people. three phases performed during SOS. first Mutualism, random selection of organisms. Second Commensalism and third Parasitism. At each phase fitness value is calculated with this formula, 1/(TUC+BW). Two casestudies using iFogsim DCNS & EAHD that give better performance than FCFS and Knapsack algorithm.

In [4], GKS algorithm is a proposed scheduling algorithm which uses knapsack for placing modules to fog devices. PEs are allocated to modules based on knapsack algorithms. GKS algorithm is used to fill a knapsack with more objects (modules) by maximizing profit and minimum weight. Profit is calculated with this formula, k1*TUC + k2*BW. This algorithm is simulated in iFogsim and improves simulation result better than FCFS, conquer and delay priority algorithm(CDPA) .

In [5], Author concentrates on VM load balancing using Bee Colony Optimization (BCO). The proposed BCO algorithm processes the VM request by considering several factors like time, historical factors and uses fixed tasks. This algorithm is developed in the Cloudsim toolkit. Simulation results show that BCO is superior in terms of reducing completion time with FCFS and dynamic load balancing algorithms.

[6] presents a workflow scheduling optimization technique. The main structure of this method is built on a tree, and it is used to plan processes in the fog environment. This algorithm has two main phases: in the first phase, all available resources are classified using the resource discovery algorithm, and then each of the sources transmits its information to the parent node that is directly connected to it, making it easier to control the resources by any parent node; in the second phase, assigning resources to perform workflow tasks is done while considering the quality of service determined by the user. This method, in general, provides a solution for scheduling processes while taking into consideration the QOS parameters specified by the client.

In [7], The researchers introduced the knapsack for task scheduling of parallel video transferring in the cloud, based on the minimal completion time (MCT). They employed the max min method, which involved calculating the most powerful computers and mapping them to a number of segments, which were then scheduled using the MCT algorithm. The findings reveal that the max min algorithm outperforms MCT in terms of execution time and segment count.

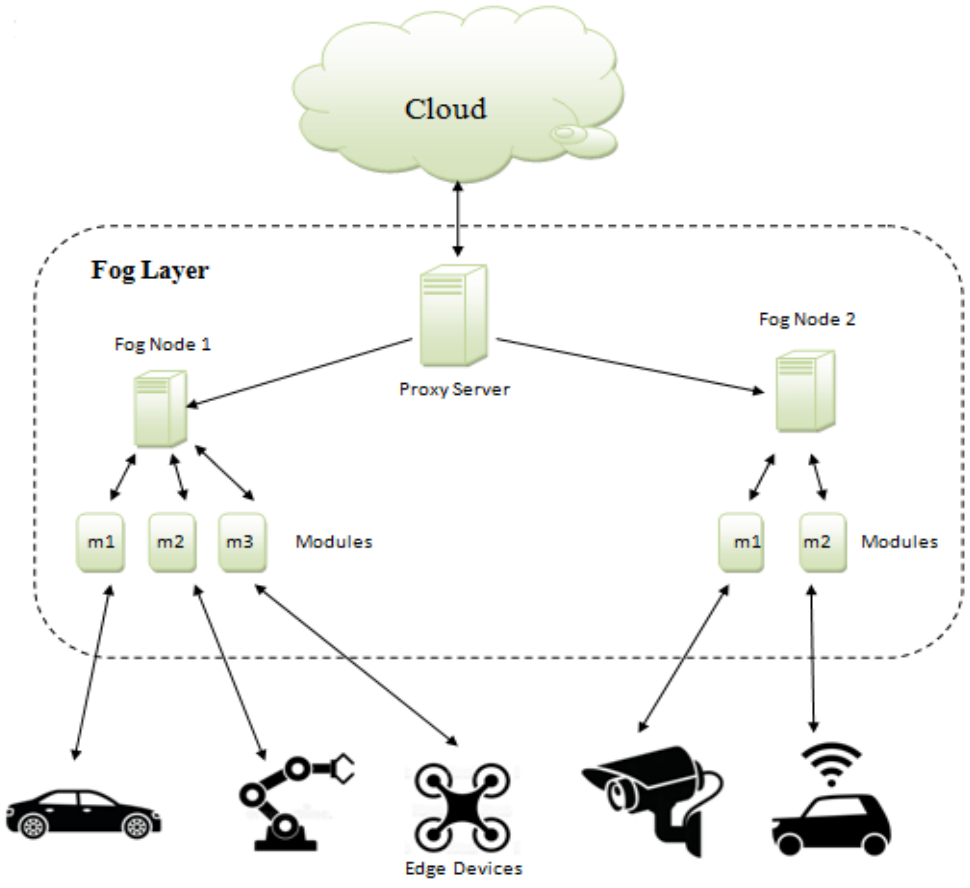
Using a knapsack and dynamic programming, the authors of [8] determined the optimal sequence of jobs to execute depending on the deadline and the least amount of money spent. This was done while taking into consideration the limited capacities of subproblems and finding the optimal settings for features such as deadlines in uncertain jobs and network congestion as well as erroneous task size.

3. Application and System Models

Fog computing is a type of cloud computing that executes applications in fog devices between end devices and the cloud. For dispersed data and low latency, we leverage this paradigm with the advantages of cloud and edge. The lower layer of the architecture contains IoT sensors, which are responsible for receiving and transmitting data through gateways to the higher layer, as well as actuators, which are responsible for system controls. Fog computing, in reality, allows edge devices to filter and analyse data. Each fog network

application has a unique topology.A graphical user interface (GUI) module of the iFogsim simulator allows you to construct custom and ready topologies [1]. This GUI can be used to add sensors, actuators, fog, cloud, and connection elements to the topology. We use a case study in iFogsim to develop a case study using the new topology for the car parking. Other modules in the simulator can read and execute these topologies.

There are many significant areas where fog computing can play a vital role in IoT applications like Connected Car, Smart Traffic Lights, Smart City, Smart Waste Management, Smart Home, Healthcare and Activity Tracking, etc. Using fog computing, traffic signals can open roadways based on the detection of flashing lights. It detects pedestrians and bikes and calculates the distance and speed of approaching cars. When it detects movement, sensor lighting switches on, and vice versa [12]. Smart traffic lights may be thought of as fog nodes that communicate with one another to convey warning signals to adjacent cars. Wi-Fi, 3G, smart traffic signals, and roadside equipment all help to optimize fog interactions between vehicles and access points [11].



Case Study 1: intelligent surveillance through distributed camera networks

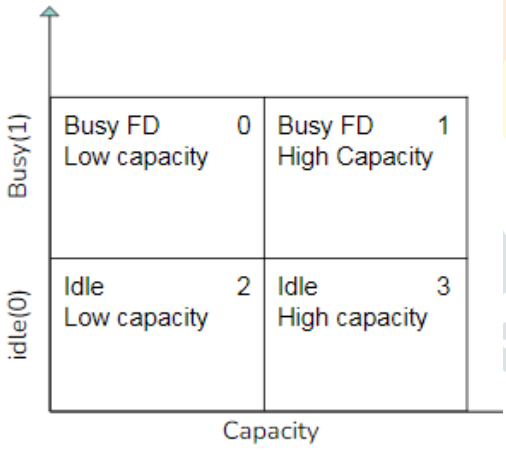
Based on a distributed system of monitoring cameras in the domains of healthcare, transportation, security and manufacturing [1], this case study is presented. In this scenario, the application model consists

of five functions. Processing of raw video to identify certain objects and movements streams in front of the camera Object tracker for the purpose of calculating an Configuration of the PTZ for maximum effectiveness. PTZ setup is used for altering the camera position. The physical camera and the actuator are both used in this application. Interface with the user for the purpose of transmitting a portion of the tracked object to the user's device The physical topology of case study-A is represented by the DCNSGame. as illustrated in Fig. 1. All cases in the dotted line are included in this architecture. Fog devices are contained within a box. m M1 is for module 1.

4. Proposed Work

4.1 Clustering approach

Fog computing model maps requested task to virtual machine according to efficient scheduling algorithm. For scheduling fog devices, we employ the k-means clustering technique. The simulator's default resource scheduler shares the fog device's resources evenly across all running application modules. K-means clustering is used to organise user tasks based on the time and cost they take to complete. Clustering simplifies the process of locating a set of VM-friendly tasks. Each virtual machine (VM) is cost-efficient.



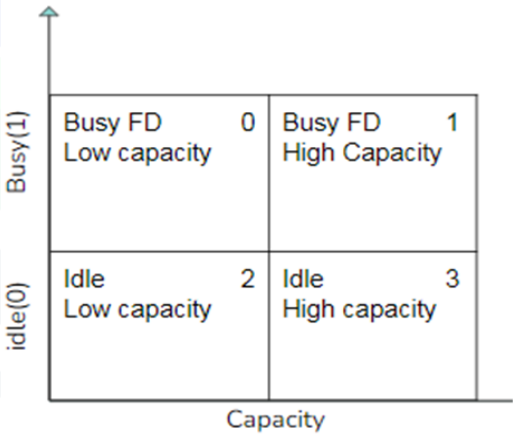


Fig.4.1 Clustering tasks based on cost and time Fig.4.2: Assigning clustered task to optimal Fog Nodes

We group tasks together based on their time duration and cost. Because we use k=4 in k-means, we get four clusters. These four clusters reflect four types of tasks, such as Cluster 0's low-cost, low-time work set (c0). Cluster 1 is a job set with a high cost and a short turnaround time (c1). Cluster 2 signifies a low cost and a long time jobs. Cluster 3 denotes a job set with a high cost and a long duration.

Clustered tasks are assigned to modules (VM). We categorize the Fog Nodes(FD) into four categories. First we check for FD is idle or bust and then we define the FD as having high capacity or low capacity.as we can see in Fig(4.2). The idea behind clustering is to find an optimal pair of tasks and VMs.

Cluster c0 having low cost and low time tasks are assigned to busy FD , which have low capacity. Busy FD can handle low cost tasks without making overhead to such small tasks. The idea behind this is to fully utilize all fog nodes and assign tasks that match the current situation of FD. Cluster c1 is assigned to a busy FD , which has high capacity power, so tasks with high cost can be handled by such FDs.Cluster c2 having low cost but high processing time is mapped to idle FD which can handle those tasks easily. Similarly cluster c3 is assigned to such idle FDs that have high capacity i.e. high computational power.Clustering makes it easy to find a set of tasks for a VM with minimum cost. Therefore, the integration of ML method i.e. Clustering in scheduling tasks in fog computing will give better quality of services(QoS) with low execution cost and low network usage.

4.2 Simulation Workflow

iFogsim is a java based simulator in which we can develop and test fog computing scenarios, topologies and applications. The flow of working in iFogsim is shown in Fig(4.3). First FogBroker class creates the fog environment, then createApplication method creates case study or any application to test fog nods, sensors, actuators in fog environment. createFogDevice method that creates Number of fog devices having different attributes, capacity. These attributes allow you to build fog devices and provide their hardware attributes, such as node name, MIPS, RAM, uplink bandwidth, downlink bandwidth, level, ratePerMips, busyPower, and idlePower.Submit application and here we are using our proposed scheduling algorithm “clustering scheduling” we call it via Clustering(Tuples, FD). This method clusters the Tasks/Tuples according to our proposed approach in Fig(4.1), after it assigns the VM to our clusters and appAllocationPolicy class plays that role for scheduling and after it we update the simulation energy, cost and network usage and result will be shown in simulation.

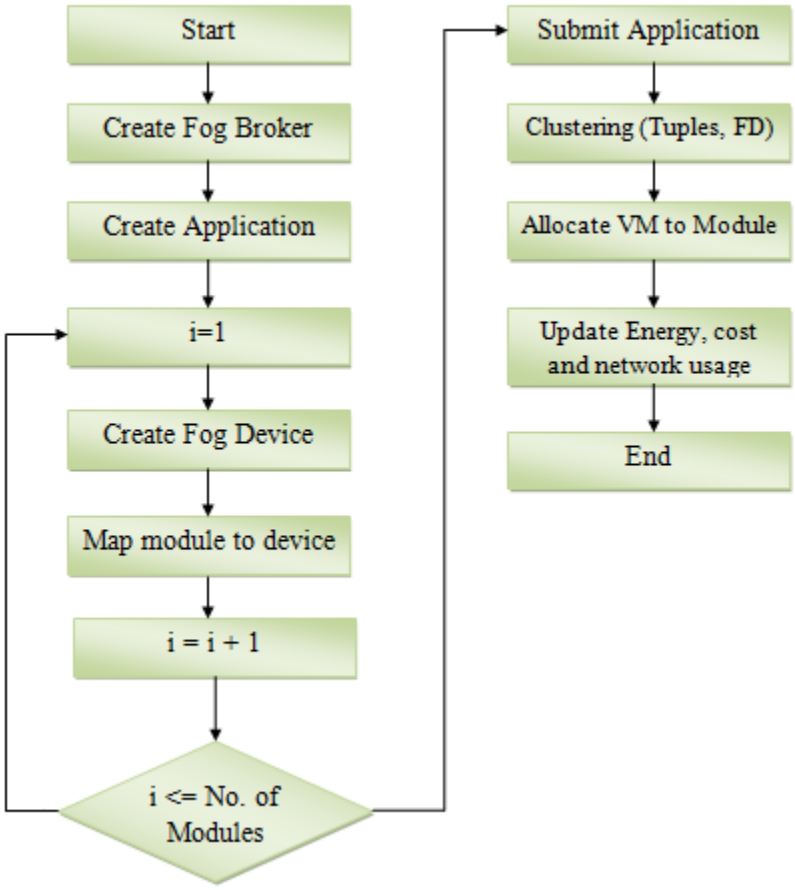


Figure 4.3: working Flow of iFogsim Application

Algorithm 1: Cluster Scheduling

Initializing VM scheduler.

k=4

call k-means(tuples) //tasks are grouped based on cost and time

FD are categorized using idle/busy state and with their capacity

c0 is assigned to f0

c1 is assigned to f1

c2 is assigned to f2

c3 is assigned to f3

In the Algorithm 2 , the fog broker and application are created and for each camera and area create a fog device. the applications add to fog broker with the creation of fog devices. The application adds to fog brokers with the creation of FD. module mapping and start of ifogsim with scheduling of modules for VM allocation by Clustering Algorithm 1.

Algorithm 2: DCNS with Cluster Scheduling

Create fog broker

Create appliaction (Modules, Edges, Tuples, Workflow)

For i=1 to Areamax do

For j=1 to Cameramax do

Create FogDevice(node name, MIPS, Ram, Storage, upper BW, lower BW, Busy power, idle power)

End for

End for

Initialize a module mapping.

Submit application

Start iFogsim

For i=1 to Fogdevicemax do

Add module to fogDevice(i)

Call cluster scheduling (Tuples,fogDevice)

Allocate PE to modules call AppModuleAllocationPolicy

Update energy cost

Stop iFogsim

We group tasks together based on their time duration and cost. Because we use k=4 in k-means, we get four clusters. These four clusters reflect four types of tasks, such as Cluster 0's low-cost, low-time work set (c0). Cluster 1 is a job set with a high cost and a short turnaround time (c1). Cluster 2 signifies a low cost and a long time jobs. Cluster 3 denotes a job set with a high cost and a long duration. Clustering makes it easy to find a set of tasks for VM with minimum cost. Therefore, the integration of ML method i.e. Clustering in scheduling tasks in fog computing will give better quality of services(QoS) with low execution cost and low network usage.

The execution cost parameter is based on following formula,

Execution Cost= CC+CT*LUT*RM*LU*TM

CC is Current Cost

CT is Current Time

LUT is Last Utilization Update TIme

RM is Rate per MIPS

LU is Last Utilization

TM is Total MIPS of hosts.

5. Experiment And Results

For this research, we used the iFogsim library to run simulations. This Java language library has modules and classes that can be used to simulate fog computing. People who use the Cloudsim library will need this package and its classes, which are called iFogsim. If we want to run the programme, we'll need to have a PC with an Intel Core i5 processor, 3 gigabytes of RAM, and Microsoft Windows 10. To run the new scheduling algorithm, we're going to use two case studies to do it.

The initialization values for iFogsim entities as following:

AppModule, MIPS is 1000, memory of 10 MegaByte, BW is 1000 KBs and ram capacity is 10.

FogDevice, Storage capacity is 1 GB, BW is 10000 KBs, Cost of processing is 3.0, cost of using memory is 0.05, Cost of using Storage is 0.001.

We run the simulation seven times for each of the two states (Number of cameras) and two methodologies (SOS and Clustering) for DCNS. Our comparison is based on the best results obtained with the identical configuration for case studies. In Table.2, we illustrate the actual execution cost and network utilization in two case studies. In this given fig we can see our proposed scheduling method DCNSCluster is Cost effective that existing Scheduling method.

Test number	DCNS SOS	DCNS SOS	DCNS Cluster	DCNS Cluster
	Execution Cost	Network Usage	Execution Cost	Network Usage
1	21790	11141	14923	6759
2	26717	11468	17849	7085
3	26468	11305	16950	7085
4	20543	10789	15962	6693
5	24831	12059	14521	6983
6	22237	11427	14944	6677
7	20498	10183	16577	7351

In Table(5.1) Seven times results of DCNS case study are collected using iFogsim, it shows the Execution cost in milliseconds (ms) and network usage in Kilobytes (kb). For comparison of existing SOS and proposed Clustering scheduling, first we take average of this results. The averaged value is shown in Table(5.2). Based on that table(5.2) we are comparing the both scheduling results in Fig(5.1) and Fig.(5.2).

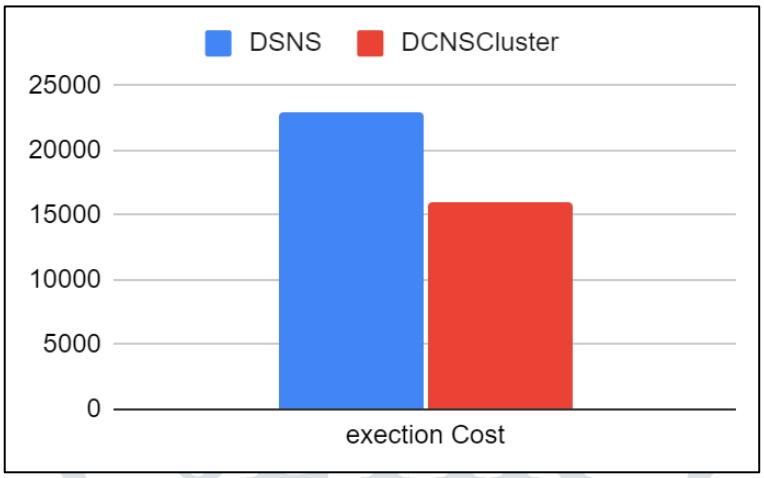


Figure 5.1: Execution cost(ms) comparison for DCNS and DCNS-Clustering

Fig(5.1) denotes the execution cost of DCNS with existing scheduling and DCNS with Cluster Scheduling. As we can easily recognize that DCNS-Cluster is having minimized execution cost then DCNS with existing scheduling (SOS).

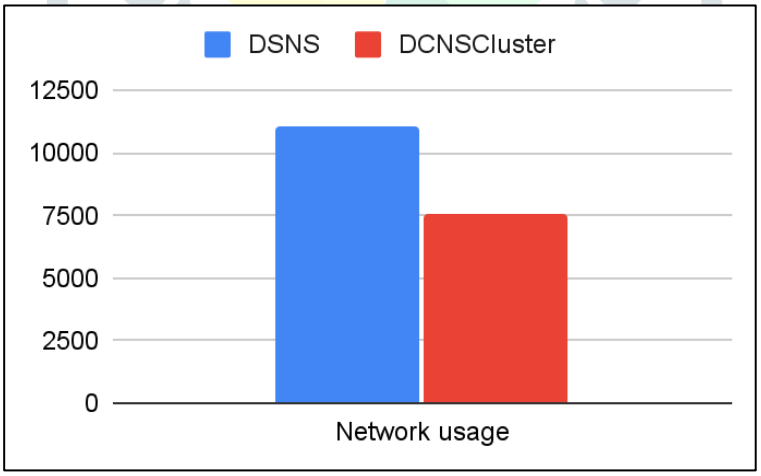


Figure 5.2: Network Usage(kb) comparison for DCNS and DCNS-Clustering

Fig(5.2) denotes the network usage of DCNS with existing scheduling and DCNS with Cluster Scheduling. As we can easily recognize that DCNS-Cluster is having minimized network usage then DCNS with existing scheduling (SOS).

The simulation result shows the DCNSClustering shows better execution cost and network usage than DSNS-SOS (exisitng Scheduling). We Run Simulation for 2 states of (camera,4) and (camera,16) with 2 scheduling methods DSNS, DCNSClustering.

Table 5.2: Averaged output for comparison of simulation results

6.

Scheduling scheme	No. of Camera	Avg. Execution Cost (ms)	Avg. Network Usage (kb)
DCNS-SOS	4	23735	11611.12
DCNS-Clustering	4	15300	7004.3
DCNS-SOS	16	2594224	664993
DCNS-Clustering	16	601114	115913.1

Conclusion

Our new scheduling method based on machine learning, such as the K-means clustering algorithm deals with resource allocation issues in fog computing. This approach follows the task clustering and optimal cluster allocation to FD for finding a better VM for allocation. This proposed algorithm improves the execution cost and reduces network traffic and improves execution time in a fog environment.The experimental evaluationof the proposed model is developed on iFogsim toolkit. The results proved that the proposed clustering algorithm is superior in terms of reducing makespan(total time taken to process a set of jobs for its complete execution) when compared to existing scheduling scheme (SOS) in iFogsim.

References

1. H. Gupta, A. V. Dastjerdi, S. K. Ghosh, and R. Buyya, “ifogsim: A toolkit for modeling and simulation of resource management techniques in internet of things, edge and fog computing environments,” arXiv preprint arXiv:1606.02007, 2016.
2. E. Ghaffari, “ Providing a new scheduling method in fog network using the ant colony algorithm”, Collection of Articles on Computer Science (2019).1 URL https://www.scipedia.com/public/Ghaffari_2019a
3. D. Rahbari and M. Nickray, "Scheduling of fog networks with optimized knapsack by symbiotic organisms search," 2017 21st Conference of Open Innovations Association (FRUCT), 2017, pp. 278-283, doi: 10.23919/FRUCT.2017.8250193.

4. Rahbari, Dadmehr& NICKRAY, MOHSEN. (2019). Low-latency and energy-efficient scheduling in fog-based IoT applications. TURKISH JOURNAL OF ELECTRICAL ENGINEERING & COMPUTER SCIENCES. 27. 1406-1427. 10.3906/elk-1810-47.
5. Mallikarjuna, Basetty& Krishna, P.. (2018). A Nature Inspired Bee Colony Optimization Model for Improving Load Balancing in Cloud computing. International Journal of Innovative Technology and Exploring Engineering. 8. 51-55.
6. Yin, Luxiu& Luo, Juan & Luo, Haibo. (2018). Tasks Scheduling and Resource Allocation in Fog Computing Based on Containers for Smart Manufacture. IEEE Transactions on Industrial Informatics. PP. 1-1. 10.1109/TII.2018.2851241.
7. F. Lao, X. Zhang and Z. Guo, "Parallelizing video transcoding using Map-Reduce-based cloud computing," 2012 IEEE International Symposium on Circuits and Systems (ISCAS), 2012, pp. 2905-2908, doi: 10.1109/ISCAS.2012.6271923.
8. Rodriguez, Maria Alejandra and RajkumarBuyya. "A Responsive Knapsack-Based Algorithm for Resource Provisioning and Scheduling of Scientific Workflows in Clouds." 2015 44th International Conference on Parallel Processing (2015): 839-848.
9. Kaarthik, K &Sridevi, Annathurai&Vivek, C. (2017). Image processing based intelligent parking system. 1-4. 10.1109/ICEICE.2017.8191876.
10. Peter, N. FOG Computing and Its Real Time Applications. Int. J. Emerg. Technol. Adv. Eng. 2015, 5, 266–269.
11. Nikoloudakis, Y.; Markakis, E.; Mastorakis, G.; Pallis, E.; Skianis, C. An NF V-powered emergency system for smart enhanced living environments. In Proceedings of the 2017 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), Berlin, Germany, 6–8 November 2017; pp. 258–263.
12. Dastjerdi, A.V.; Buyya, R. Fog Computing: Helping the Internet of Things Realize Its Potential. IEEE Comput.Soc. 2016, 112–116. [CrossRef]
13. "Fog Computing - GeeksforGeeks," GeeksforGeeks, Apr. 29, 2020. https://www.geeksforgeeks.org/fog-computing/ (accessed May 02, 2022).
14. "Fog Computing and the Internet of Things: Extend the Cloud to Where the Things Are."http://www.cisco.com/c/dam/en_us/solutions/trends/iot/docs/computing-overview.pdf (accessed May 02, 2022).
15. SaikatDutt, S. Chandramouli, and Amit Das, MACHINE LEARNING. Pearson, 2019