MTFCT: A task offloading approach for fog computing and cloud computing

Rajni Jindal
Deprtment of CSE
Delhi Technological University
Delhi, India

Neetesh Kumar
ABV-Indian Institute of Information Technology and Management
Gwalior, Madhya Pradesh, India

Hitesh Nirwan
Department of CSE
Delhi Technological University
Delhi, India

Abstract— **Cloud computing is an important computing paradigm for handling all types of computations, even the smaller ones in the past. But sometimes, it becomes ineffective when the task is to be done in real-time, with very low latency. Therefore, fog computing was introduced as a supplement computing paradigm for cloud computing. Internet of Things based applications perform better with the amalgamation of its and the fog computing. Due to low capacity, when fog can't compute the task on its own, heavy computations are offloaded from fog to cloud. But when to offload the task from fog to cloud is a major decision. The decision is to be made out to offload the tasks from fog to cloud is very crucial, so this paper presents an idea to solve this problem.**

Keywords— Cloud Computing; Fog Computing; Task Offloading

I. INTRODUCTION

With the advancement in the technology area, more and more electronically enabled devices, which are capable are connecting with the internet, so the Internet of Things (IoT) devices are increasing day by day. These devices are communicating with each other, so basically, they are generating lots of data [1].

Because of this, the burden on cloud computing to solve computations and to provide the result back in a given time was also increasing day by day.

Cloud computing is an important computing paradigm to handle heavy scientific workflows by providing infinite virtual resources but doesn't work well where latency requirement is low [2].

There may be a case where latency-sensitive applications demand results back in real-time. Cloud may provide an accurate result, but it may return the result late, which may diminish the whole objective.

To solve this issue, fog computing was introduced in 2014 [3]. Fog computing is an online computing paradigm to solve low complexity problems in real-time. It was introduced as a supplement to cloud computing. Its aim was never to replace cloud computing, rather reduce the overload on cloud datacenters and provide fast services to the latency-sensitive application [14].

So it can perform both the operations, storing as well as processing the data near the edge of the network or near the end-user. The interconnection of IoT devices with fog nodes and fog with the cloud is shown in fig 1.

Fig. 1. Three tiers IoT, Fog and Cloud Architecture [11]

Fog computing is decentralized in nature, unlike cloud, which is centralized [4]. It also provides fog computing an upper edge over cloud computing

II. BACKGROUND KNOWLEDGE AND MOTIVATION

According to an estimate, more than 50 billion devices will be connected to the internet by 2020 [5]. The majority of these devices are not able to process their data on their own. So, we need the help of fog computing also along with cloud computing. IoT is the future, but we need to amalgamate it with fog computing. We can not overload cloud datacenters.

Moreover, providing a reliable result is still not an easy task. The performance of fog computing is dependent upon fog nodes placement as well as the resources provided to the fog nodes. Fog nodes are generally much closer, at the edge of the network [6]. Therefore, their latency time is very less as compared to cloud datacenters.

Fog nodes are small data centers that provide the computational capability to these latency-sensitive devices [7]. They are generally placed between the end devices and the cloud data centers in the network — the closeness of these nodes with end devices benefits in providing low latency [8].

The drawback of these fog nodes is their limited capacity. They cannot provide high computational and storage power [9]. Because of this reason, some data is offloaded from these fog nodes to the cloud data center for processing. ‘

Fig 2. Explains the architecture of fog and cloud. From the figure, we can observe that there are three levels. At the lowest level, we have sensors and actuators. Sensors gather the data from the surrounding, and this data is in the form of data streams known as tuples.

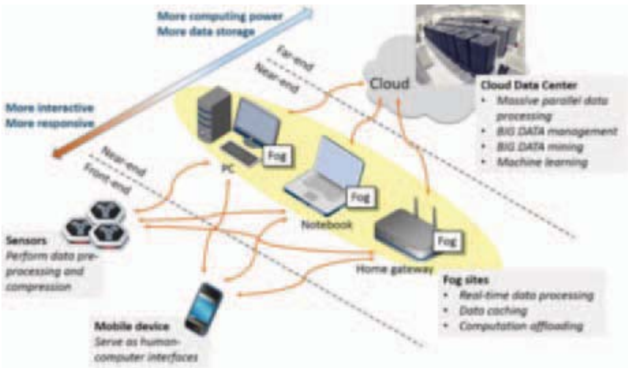

Fig. 2. Cloud/Fog Computing Architecture [10]

While actuators receive that output data from the upper layer, these sensors and actuators work as the front end. This layer is in the shortest distance with the user. This data is then transferred from this front end to near end devices, a sensor to the fog nodes, which are available in the fog layer (middle layer). These devices process the data according to the requirements and send the result back to the actuators.

If the data is not fully processed, the processed data will be offloaded to the cloud for further processing and storage purposes. Cloud may lead to a delay in the application result but will complete the job.

After completing the task, the result will be sent back to the actuators. This cloud layer is generally used for handling big applications and also for the storage purpose.

There is a various application where fog is integrated with cloud and provide support for various IoT applications very effectively like in the smart home, smart transportation, smart healthcare sytem, visual security, education sector, etc.

Aim of this is to provide a better computing experience, which can support all types of applications, even the latency-sensitive one too.

Some of the advantages of fog computing over cloud computing are low power consumption, low latency, bandwidth saving, decentralized nature, security, mobility, etc.

Location awareness is also high in fog computing in comparison to the cloud. Physically fog nodes are decentralized and distributed, while the cloud is centralized. It gives an upper edge to the fog computing and fog nodes. They can be well distributed to decrease the load on any fog node.

These advantages encourage the user to move to fog computing. As the fog layer is sharing part of the original data which it received from the terminal nodes, it is decreasing the overall bandwidth requirement, which is very useful [12].

Fog computing has some disadvantages also. These are computation and storage power of fog nodes that are much less than the cloud datacenter. But in a real environment, it does not affect much.

As the original purpose of fog computing is to provide small computation and storage facility to the end-user. So heavy computational requirement is not needed to the end-user.

In offloading, the task is to contract out to the external agents to handle. External resources work towards the completion of the intended task [13]. It may be due to the high computational requirement or storage requirement, which task source or mobile device is unable to perform.

The decision when to offload the data to the cloud is an important one. It can affect the latency significantly. If the decision when to offload the data is not taken place in time, it can degrade the whole computing experience and can affect the latency in a very negative matter.

If we upload the data to the cloud, when it was not needed, latency will be increased drastically. If we don’t upload the data to the cloud or upload late, then also the latency will be increased. We have worked in this direction and proposed a new algorithm to solve this offloading issue.

III. OFFLOADING APPROACH

A. Problem description

We are assuming that there is only one cloud datacenter, which will provide all the heavy computing facilities. It is located at a far distance from the end-user. Bandwidth between cloud and fog nodes (B_{FC}), cloud datacenter, and terminal nodes/ end devices (B_{EC}) are already known in Mbps. The energy required to transfer each unit of data from fog node to cloud datacenter (E_{FC}) and from end devices to cloud datacenter (E_{EC}) is also already known.

There is N number of fog nodes in an area with each having different processing capacity (P_N) and different pricing for each data unit (FP_N). Pricing for cloud data center varies according to the chosen VM. So, the task by end devices can be configured as T_N = {T_1, $T_{2,....,}$ T_N}. It will be decided whether the fog nodes are chosen or cloud data center or both.

Bandwidth requirement to offload the data to the cloud is generally greater than the bandwidth requirement to offload the data to the fog. Also, the prices to lease the cloud datacenter resources and their capacity is much larger than the prices and capacity of fog nodes.

Total work is done in three phases. In phase 1, it is decided that whether the generated task by an end device can be processed by the end device itself or not.

If the end device is not capable of doing so, only in that condition, the decision to process the data through fog nodes will be taken. Otherwise, offloading will not be performed.

Now, if the fog node can process the data and also within the given allotted time, then the task will be processed by the fog node. Otherwise, data will be offloaded to the cloud datacenter.

There will be a central fog manager, who will be aware of all the fog node processing power and their impact zone, where they provide services.

Fog manager will select the best fog node to provide service to mobile devices based upon its computation capability and its impact zone. Tasks that fog node can not process on their own will be offloaded to the cloud datacenter.

To determine whether offloading the task to fog node or cloud from mobile devices are in our favor or not, we have proposed Mobile to Fog and Cloud Transfer (MTFCT) algorithm.

B. Execution Time

Execution time is divided into many phases, which are the following:

1) T_{OD} = A decision is made before transferring data to the fog node, whether the fog nodes can process the given task or not. If the fog nodes meet the task requirement, the task will be given to the fog nodes; otherwise, the task will be given to the cloud datacenter.
This decision is based on the task processing requirement and fog nodes capacity in the vicinity of the fog node manager. This decision takes time, and it is known as $T_{OD.}$ It is a crucial decision. It will take place with the help of various other parameters.

2) T_{MF} = Task size*Time required to transfer one unit of data to the fog (Time required to transfer whole data from mobile device to the fog node)
The selection of fog node will be managed by the fog manager. This time will only be considered when the fog manager decides that the given task can be processed by the fog node in the given time. It depends upon the available bandwidth and task size.

3) T_{MC} = Task size*Time required to transfer one unit of data to the cloud data-center (Time required to transfer whole data from mobile device to the cloud data-center). This time is generally greater than $T_{MF.}$

4) T_{EF} = Task size/fog node processing capacity (Time required to execute a task by fog node). If the task is processed by the fog node, it is calculated using the task size and processing capacity of that particular node. Lesser the processing capacity higher will be the $T_{EF.}$

5) T_{FM} = Time required to send the result back to the mobile device. The same channel is used as in T_{MF}. This time will depend upon bandwidth and result in size. The more the task size more will be its value.

6) T_{EC} = Tasksize/cloud datacenter processing capacity If the task is processed by the cloud datacenter (Time required to execute a task by cloud datacenter). It is calculated using task size and processing capacity of that cloud datacenter, which is generally very high. Its value is always less than $T_{FM.}$

7) $T_{FC=}$ Time required to send the task from fog to cloud. Sometimes a task is partially processed by a fog node and is sent to the cloud data center for further processing. In this, the remaining task size and bandwidth between fog node and cloud datacenter play an important role. So this time is known as $T_{FC.}$

8) $T_{CF=}$ After the task processing, if the result is sent back to the fog node, the same channel is used as in $T_{FC.}$ So this is known as $T_{CF.}$

9) $T_{CM=}$ After the task processing, if the result is sent back to the mobile device, the same channel is used as in T_{CM}. This time will depend upon bandwidth and result in size.

10) T_M = Execution time if the task runs on mobile devices itself.

So total execution time a task requires can be anything based upon a combination of the factors mentioned above. If the cloud data center gets involved, the execution time will get high; otherwise, if only fog nodes are involved, the value of execution time will be very less.

C. Cost of Execution

We can divide our cost of execution on a remote center into two parts; the cost of execution on cloud (C_C) and the cost of execution on fog (C_F). Cost of execution on fog can be further divided into two parts which are following;

$$C_F = C_{EF} + C_{EC} \quad (1)$$

(C_{EC} depends upon how much part of the task is sent for processing by the local fog node to the cloud node)
Where C_{EF} is a cost for executing the part of the task on fog node only, and C_{EC} is the cost for the part sent to the cloud datacenter from fog node for further processing and other purposes. If the task is processed fully by the fog nodes themselves, the value of C_{EC} will become zero. So in that case,

$$C_F = C_{EF}$$

C_M = cost to run the task on mobile devices itself.
If we don't offload our task to the cloud or fog node, in that case, we calculate this value. This is generally used to check whether it is beneficial to offload the task or not. If it is not beneficial, the task will not be offloaded. But it is not the only

factor which decides; other factors are also there like execution time, energy requirement, etc.

Generally, the cost of processing on cloud takes high value, but still, some tasks can not be processed locally by fog nodes or the mobile devices itself, fully because of its size and required time.

So, in that case, offloading the task to the cloud datacenter becomes necessary. C_{EC} will also be affected by the scheduling policy chosen for its task scheduling.

D. Energy Requirement

Energy requirements can also be divided into two parts; E_F and E_C.

$E = E_F + E_C$. (2)

$E_F = E_{MF}+E_{EF}+E_{FC}+ E_{EC} +E_{CF}+ E_{FM}$ (3)

$E_C = E_{MC}+E_{EC}+E_{CM}$ (4)

Where,

E_F = Total energy required if task go to the fog nodes for processing. Task can also be partially executed on the fog nodes too and rest can be sent to cloud. In this case value of E_F will get vey low, because only a part of task is executing on fog node irrespective of full task.

The value of total energy consumption (E) in equation number two becomes zero if a task is decided to be processed by the end device (mobile device) itself.

E_{MF} = Task size*Energy required to transfer one unit of data (Energy required to transfer whole data to fog node)

E_{EF} = Task size*Energy required to process one unit of data by fog node (Energy required to process whole data by fog node)

E_{FC} = Energy required to transfer the remaining task for further processing by the cloud datacenter. Its value will become zero if the task is fully processed by the fog nodes themselves.

E_{CF} = Energy required to transfer the result to fog node by cloud. Its value will become zero if the task is fully processed by the fog nodes themselves.

E_{FM} = Energy required to transfer the result to the mobile device by fog node

E_C = Total energy required if the task goes to the cloud data center for processing. Task can also be partially executed on the cloud datacenter too. Its value will become zero if the task is fully processed by the fog nodes themselves.

E_{MC} = Task size*Energy required to transfer one unit of data to the cloud datacenter (Energy required to transfer whole data to the cloud datacenter). This is generally higher than the E_{MF} as the cloud data center is generally very far located as compared to the fog nodes.

E_{EC} = Task size*Energy required to process one unit of data by cloud datacenter (Energy required to process whole data by cloud datacenter)

E_{CM} = Energy required to transfer the result to the mobile device by cloud datacenter.

E_M = Energy required to run the task on the mobile device itself

E. Proposed algorithm

Algorithm 1 MTFCT algorithm

Input: parameters like T_{OD}, T_{MF}, T_{EF}, T_{FC}, T_{EC}, T_{CF}, T_{FM}, T_{CM}, T_{MC}, T_M.

Output: Offloading decision, wheather to offload to fog, cloud or not

1: Calculate the value of each variable like T_{OD}, T_{MF}, etc.

2: Check how much part of the task can be executed on local fog node too.

3: Calculate $T_F = T_{OD}+T_{MF}+T_{EF}+T_{FC}+T_{EC}+T_{CF}+T_{FM}$ (T_{EC} depends upon how much part of task is sent to cloud for processing) (5)

4: Calculate $T_C=T_{OD}+T_{MC}+T_{EC}+T_{CM}$ (6)

5: Compare T_M with T_F and T_C.

5.1: If $T_M > T_F$ and $T_M < T_C$, Task will be uploaded to the fog node

5.2: If $T_M > T_F$ and $T_M > T_C$, check if $T_F > T_C$, If yes Task will be uploaded to the cloud data center otherwise to the fog node

5.3: If $T_M < T_F$ and $T_M > T_C$, The task will be uploaded to the cloud datacenter

5.4: If $T_M <= T_F$ and $T_M <= T_C$, The task will not be offloaded to either fog node or cloud datacenter

6: Similarly, compare C_M with C_F and C_C

7: Also, calculate E_F and E_C

8: Compare it with E_M

9: Based upon the above comparisons and the weighted value of each factor (execution time, cost & energy), we decide to offload the task from mobile device to fog node or cloud node is beneficial or not take decision based upon that. (Based upon how much value or importance is given to each factor by user)

IV. Conclusion

Fog computing is the lifeline of IoT in today's world scenario. IoT can be implemented without fog computing, too, but with the help of fog computing, the efficiency of IoT increases considerably. So to achieve very high efficiency, we integrate fog computing with cloud computing.

The energy requirement of fog is very less as compared to the energy required to process the same task in a cloud datacenter. This can help us in reducing the total co2 emission to save the environment too.

In this paper, we proposed a method for task offloading to fog nodes and cloud datacenter using various parameters. In this, the involvement of users is also very high.

This is a theoretically proposed algorithm, so in the future, we would like to implement this proposed algorithm in a real fog computing environment or in a simulated environment using FogSim simulation toolkit and also provide node to node migration of task in fog computing environment and along with a better way to handle the task on cloud datacenter whether if the task is offloaded from a mobile device or any fog node

REFERENCES

[1] S. Zahra, M. Alam, Q. Javaid, A. Wahid, N. Javaid, S.U.R. Malik and M.K. Khan, “Fog computing over IoT: A secure deployment and formal verification,” IEEE Access 5 (2017): 27132-27144.

[2] M.A. Rodrigue and R. Buyya, “A taxonomy and survey on scheduling algorithms for scientific workflows in IaaS cloud computing environments,” Concurrency and Computation: Practice and Experience, 29(8), 2017.

[3] H.A. Khattak, H. Arshad, S.U. Islam, G. Ahmed, S. Jabbar, A.M. Sharif and S. Khalid, “Utilization and load balancing in fog servers for health applications." EURASIP Journal on Wireless Communications and Networking 2019.1 (2019): 91.

[4] S. Khan, S. Parkinson and Y. Qin, "Fog computing security: a review of current applications and security solutions," Journal of Cloud Computing 6.1 (2017): 19.

[5] D. Poola, M.A. Salehi, K. Ramamohanarao and R. Buyya, “Fog computing: A taxonomy, survey and future directions,” Internet of Everything (pp. 103-130). Springer, 2016.

[6] C. Puliafito, E. Mingozzi, F. Longo, A. Puliafito and O. Rana, “ Fog computing for the Internet of Things: A Survey,” ACM Transactions on Internet Technology, Vol. 19, No. 2, Article 18, 2019 .

[7] A. Mohammad, 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. IEEE, 2015.

[8] S. Delfin, S.P. Sivasanker, N. Raj and A. Anand, “Fog computing: A new era of cloud computing." 2019 3rd International Conference on Computing Methodologies and Communication (ICCMC). IEEE, 2019.

[9] F. Haouari, R. Faraj and J.M. Alja’am, “Fog computing potentials, applications, andcChallenges," 2018 International Conference on Computer and Applications (ICCA). IEEE, 2018.

[10] J. K. Zao, T. T. Gan, C. K. You, C. E. Chung, Y. T. Wang, S.J.R. Méndez, et al., “Pervasive brain monitoring and data sharing based on multi-tier distributed computing and linked data technology.” Frontiers in human neuroscience, 8, 370, 2014.

[11] R. Jindal, N. Kumar, and H. Nirwan, “A survey on cloud to fog evolution” unpublished.

[12] Y. Sun, and N. Zhang, “A resource-sharing model based on a repeated game in fog computing.” Saudi journal of biological sciences, 24(3), 687-694, 2017.

[13] M. Aazam, S. Zeadally, and K. A. Harras. "Offloading in fog computing for IoT: Review, enabling technologies, and research opportunities." Future Generation Computer Systems 87: 278-289,2018.

[14] A. Khakimov, A. Muthanna, and M. S. A. Muthanna. "Study of fog computing structure." IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus). IEEE, 2018.