The Optimization Algorithm for Load Balancing in Cloud Computing

Sarita Sharma, Dr.Mani Goyal
Department of Computer Science and Engineering, MMEC Maharishi Markandeshwar (Deemed to be University), Mullana-Ambala, India

***Abstract*—** Cloud computing is often referred to as a model that provides limitless computing services with a pay-as-you-go model. Modern cloud infrastructures provide resources as the VMs to physical machines using virtualization technology. Every virtual machine focuses on running its own operating system and it leads to utilize resources from its host physical machine (PM). For load balancing, Cloud is capable of migrating VMs from PMs which have heavy load to the ones which have light load. The objective of this process is to use the resources of a physical machine below certain threshold. Uncertainty can be a major reason of the overloading of virtual machines. Previously proposed load balancing method used genetic algorithm for the migration of the virtual machine. The delay of this algorithm increases in the network as virtual machines are migrated. Our proposed work is more appropriate than the current techniques work as it puts forward anadvancement in existing genetic algorithm for effective VM migration and better load management. The achieved results are compared against the outcomes of the previous genetic algorithm. The introduced genetic approach is evaluated over three performance parameters including delay, energy consumed, cost and VM migration.

Keywords: **Cloud Computing, Genetic Algorithm, Virtual Machine, Migration.**

I. INTRODUCTION

The popularity of CC is increasing with every passing day due to its reliance on pay-as-you-go model to deliver countless computing services. At present, cloud infrastructure provides resources to physical machines (PMs) as virtual machines (VMs) by employing virtualization technique. The users generate VMs whose utilization is done on the cloud as per the demand. Every virtual machine has its specific OS and uses resources such as Central processing Unit and bandwidth from its physical machine serving as the host. Many organizations have switched to cloud computing to fulfill its requests. Hence, the consumers and businesses do not need to install applications to use them, and can get the access to their personalized data on any computer over the web [1]. The cloud-based services are generally divided into 3 kinds namely IaaS, PaaS and SaaS. IaaS model provides access to basic resources. In Past, the runtime atmosphere for applications, development and distribution tools can be accessed. Finally, SaaS model can provide software applications in the form of a service to the end client. In cloud computing, the virtualization of every component of the hardware model is performed into virtual entities. A virtualization technique allows to running various OSs (Operating Systems) over a single PM (Physical Machine). These operating systems are differentiated from each other and the fundamental physical system through a special middleware abstraction termed as virtual machine (VM).The software that is liable to manage these numerous VMs on physical machine is designated as VM kernel [2]. The growing number of users present a number of challenges before Cloud computing as the demand for resource sharing and utilization is increasing rapidly. Hence, balancing loads between resources is a significant challenge. The existing load balancing schemes select virtual machines to migrate and find the most appropriate destination physical machines by integrating the use of various resources. They pre-determine a weight (or provide the same weight) per resource, compute the weighted product of various resource uses to denote the load of physical machines. They also represent the capacity of physical machines using the weighted product of preserved volume of every resource and then the VM is migrated from the highest loaded physical machine to the lightest loaded physical machine. In these techniques, the same or already defined weight is assigned to different resources to ignore the unique features of time-varying clouds and differently overused resources in various physical machines [3]. The virtual machines of a cloud provide different types of services using many resources which results in different overused resources and different scales of resources in diverse physical machines. The CC environment balances the load at two levels. The level of VM is mapped among applications which are loaded in cloud on VM. The load balancer allocates the demanded VM to PMs to create load balance amongst the

numerous applications from the physical machine. At lost level, the VMs (Virtual Machines) and host resources are mapped so that many arriving application requests can be processed. The algorithms to balance the load in CC are of various types. The RR is a simple LB algorithm which is based on the idea of a quantum of time or interval. This algorithm divides time into many sectors, and assigns a particular time period to every node. The node is bounded to perform its functions in the given time slot [4]. In round robin, time quantum scheduling plays an extremely crucial role, because when the time slot is too huge, the RR technique behaves similar to the FCFS algorithm. The drawback of this method is that an extra load is produced on the scheduler to find out the quantum size despite its simplicity. Also, this algorithm has more context switches which results in maximizing the turn round time and alleviating the throughput. The AMLB is dynamic in nature. In this algorithm, the information of every VM (virtual machine) along with the currently assigned requests to each VM are stored. If a new virtual machine distributes the request in the presence of multiple virtual machines, the selection of initially identified VM is done [5]. The identifier of virtual machine is returned to the manager of DC using Active Monitoring Load Balancing algorithm. The manager of DC informs this algorithm regarding the novel allocation and concentrates on forwarding a request to a known VM. The drawback of this approach is that it always discovers the minimal loaded Virtual Machine to allot a fresh arriving request, but inefficient of verifying its implementation. In the Min-Min algorithm, the tasks that required less time are accomplished at first. Thereafter, all tasks select the minimal value. The task is scheduled on the machine depending upon the minimal time. Similarly, other tasks are deleted from the task list after updating on the machine [6]. This procedure keeps on going till the final task is assigned. This algorithm is performed efficiently, in case of huge number of smaller tasks in comparison with bigger tasks. Unlike the Min-Min, the Max-Min algorithm assists in choosing the maximal value when the minimal execution time is searched. Subsequently, the machine is utilized to schedule the task on the basis of maximal time slot. Maximum to Minimum load balancing algorithm is different from minimum-to-minimum approach in just one manner. This algorithm selects maximal value after searching out the minimal execution time [7]. Afterward, the task is scheduled over the machine on the basis of maximal time slot. The assigned task is deleted from the list once the execution time of every task is updated. ACO based load balancing algorithm firstly selects a head node. This node is selected in such a way that it comprises maximum no. of neighboring nodes. The movement of ants occurs in two directions. In the first type of movement, ants move forward in a cloud to collect information about the load of nodes. In backward movement, an ant moves backward and reallocates the load among the nodes in group after finding an under-loaded or overloaded node on its route.

II. LITERATURE REVIEW

Dalia Abdulkareem Shafiq, et.al (2021) presented a new algorithmic approach for resource optimization and improving load balancing in terms of QoS (Quality of Service), prioritization of virtual machines and the distribution of resources [8]. The presented algorithmic approach was concerned with improving the use and distribution of cloud assets and minimizing the time consumed in task scheduling so that the performance of cloud applications could be improved. The new load balancing approach dealt with the above mention problems and tried to fill the gap in the existing literature works. The tested outcomes revealed that the new load balancing scheme used 78% of resources in average which was comparatively lesser than the resources used by the DLB (Dynamic Load Balancing) scheme. The new scheme also performed better with respect to makespan, and execution time.

Sreelakshmi, et.al (2019) presented a multi-objective PSO algorithm for the scheduling of tasks [9]. The main aim of this approach was to reduce makespan time, deadline along with communication overhead. The presented technique was focused on task scheduling for the balancing of loads to allocate the arriving traffic efficiently amid the back-end servers. This approach scheduled tasks by considering communication overhead, makespan and deadline. This work used CloudSim software for the simulation of the presented approach. In the simulation results, the presented scheme outperformed its rivalry schemes by reducing makespan time, communication overhead and completed the task within deadline.

Lung-Hsuan Hung, et.al (2021) proposed and combined two schemes based on genetics. Initially, the performing models of VMs (virtual machines) were derived from their building features and the associated performance computed in a cloud computing scenario [10]. This work implemented GEP (Gene expression programming) for constructing SRM (Symbolic Regression Models) that not only described the behavior of VMs but also predicted the loadings of VMHs followed by load balancing. Next, the GA considered the existing and future loads of the VMH using the VMH load evaluated by the GEP, and decided an optimum VM-VMH task to migrate and load-balance the VM. This work estimated the performance of the introduced technique by applying it in a practical cloud-computing scenario called Jnet. In the tested results, the presented scheme performed superior to the

existing schemes.

Ronak Agarwal, et.al (2020) presented PSO algorithm based on mutation for balancing load amongst data centers [11]. Minimizing MakeSpan and improving fitness function was the main aim of the presented algorithmic approach. The mutations applied to the optimum solution provided by the current PSO algorithm improved MakeSpan and fitness function. In contrast to PSO algorithm, the MPSO algorithm provided better outcomes. The comparison of the presented technique was performed in terms of MakeSpan feature. The presented technique performed load balancing by scheduling VMs (Virtual Machines) pre-emptively. The future work will be focused on considering several other parameters such as resource usage, throughput, waiting period, average time, etc. to perform load balancing.

Vishalika, et.al (2018) presented a new framework called LD_ASG (load-based task assignment algorithm). It involved assigning tasks to the selection methods i.e. the virtual machine with minimal load for the efficient usage of resources [12]. Simulation was conducted to change the number of VMs (Virtual Machines) to check the performance of the presented scheme by using resources optimally. In the test results, the presented scheme selected the virtual machines with minimal load in all settings and improved the exploitation of resources. The presented scheme had proved its appropriateness in load balancing of VMs to save cloud system from being overloaded.

Zhao Tong, et.al (2020) put forward a new DLB task scheduling algorithm on the basis of Deep reinforcement learning under the limitations of SLA (Service Level Agreement) to decrease the imbalanced loads and task rejection rate of virtual machines (VMs) [13]. This work initially adopted the DRL technique to choose the appropriate Virtual Machine for the task and then determined whether executing the task on the chosen virtual machine violated the SLA constraints. In case of SLA violation, the task was rejected and the response was a negative result to train Deep reinforcement learning; in other case, the task was obtained and executed, and the VM returned a response as per the balancing of the load followed by the implementation of the task. In contrast to the 3 other schemes implemented to the created standard at random and the Google real user workload trace standard, the suggested approach was appropriate to balance the load of VMs and reduce the task rejection rate; thereby it enhanced the general scale of cloud computing applications.

Dr. Mani goyal ,Avinash Sharma et.al(2020) presented a leveraging remote cloud services enhances resource efficiency but is highly dependent on network quality. Optimal performance requires intelligent task division between device and cloud based on real-time conditions. Secure data distribution using hybrid encryption ensures privacy, making dynamic cloud frameworks vital for reliability in poor connectivity scenarios.

Dr. Mani goyal,Avinash Sharma et.al(2019) are implements Efficient cloud use relies on network quality and smart task division. Hybrid encryption ensures secure data sharing, while dynamic frameworks optimize performance in real-time, especially under poor connectivity conditions.

Dr. Mani goyal ,Avinash Sharma et.al(2022) improved and combines Enhancing Hybrid Encryption Techniques for Secured Data Processing for Small Medium Enterprises in cloud. Authors study proposed cryptographic methods to ensure data integrity and confidentiality. By comparing DES, AES, Paillier, and Blowfish algorithms, we evaluated performance based on file size, processing time, and memory usage across various file formats.

III .RESEARCH METHODOLOGY

To deal with node failure related challenges in cloud networks, this research project puts forward a meta-heuristic algorithm known asBFO (Butterfly Optimization). The proposed algorithm consists of many nodes. Depending on the failure rate and the shortest execution time, a candidate node is chosen from among all these nodes. This case makes use of a master node to set a threshold value.This threshold value includes two parameters. These are the highest execution time and failure rate. Nodes that have less than or equal to failure rate and the minimal execution time are opted by the master node as candidate nodes. Compared to the threshold value, node N1 has a lower value.Thisis the main reason of selecting this node as the candidate node. Node N2 contains one low and one high parameter. Therefore, this node can’t be selected as a candidate node. A node N3 is selected as the candidate node because it contains a value equal to the threshold. Similarly,it is not feasible to select another node N4 as candidate node because its value is greater as opposed to the threshold level. Post its selection, the candidate node starts performing its task. In this case, various tasks are initiated. A node moves from its place post the task completion. Hence, resulting in task failure. To overcome the issue of failure occurrence due to node movement, a new methodology is put forward in this work. The envisioned algorithm inducts a new parameter known as master node duration. The finalduration to add end users is referred to as master node time which assists in node collaboration. Given the following formula for computing the master node time:

1. E-cost= maximal execution time + Time required by master node (master node time)

Then, we will compute each node's profit.

2. Profit of each node = E-cost+ Failure time of each node

3. Weight of each node= No. of tasks + maximal execution time/Profit

The node with highest weight is selected. The provided formula is used for the weight measurement.

The envisioned algorithm takes the following steps:

Step 1: Get list of all VMs working on all hosts.

Step 2: Initialize no migration is performed.

Step 3: Get resource consumption, failure rate, and execution time of all machines.

Step 4: Built transition matrix for hosts and VMs.

Step 5: Loop will execute until all machines on over utilized hosts are migrated.

Step 5.1: Calculate the current utilization of each host for that particular VM that needs migration.

Step 5.2: Check creation history of the VM.

Step 5.3: Compare increase in utilization of selected hosts with other hosts.

Step 5.4: Select host for which increase in utilization is minimum End loop

Step 5.6: If maximum utilization exceeds upper utilization threshold go to step 5.1.

Step 6: Else choose that particular host for migration.

Step 7: return migration List

END

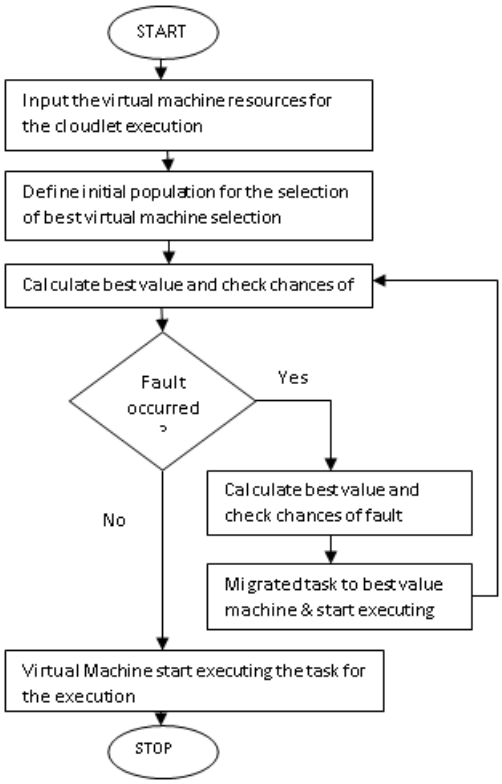

Figure 1: Proposed Flowchart

IV. RESULT AND DISCUSSION

MATLAB simulator is used to implement the proposed algorithm as in realistic cases, its complexity is high. Depending on power consumption and execution time, the comparison among the performance of proposed and existing algorithms is evaluated. Table 1 features the simulation parameters used in this research.

Number of VM	10
Number of cloudlets	60
Host Memory	2 GB
Processor	Xenon
Number of Data centers	5

Table 1: Simulation Parameters

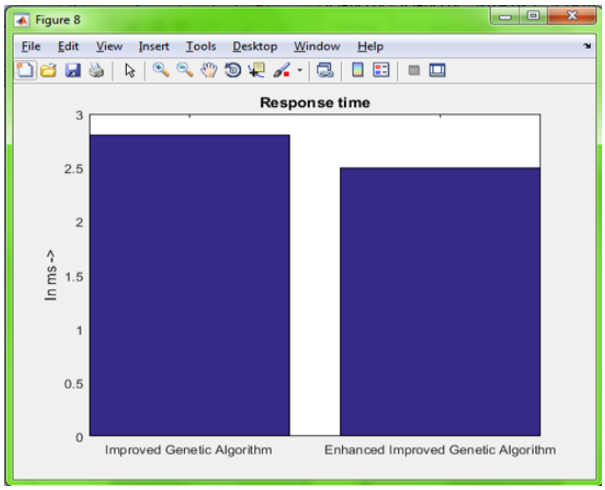

Fig 1: Comparison graph of Response Time

Figure 1 shows the response time of the improved genetic algorithm and proposed enhanced improved genetic algorithm compared for the performance analysis. The response time of enhanced improved genetic algorithm is less as compared to improved genetic algorithm.

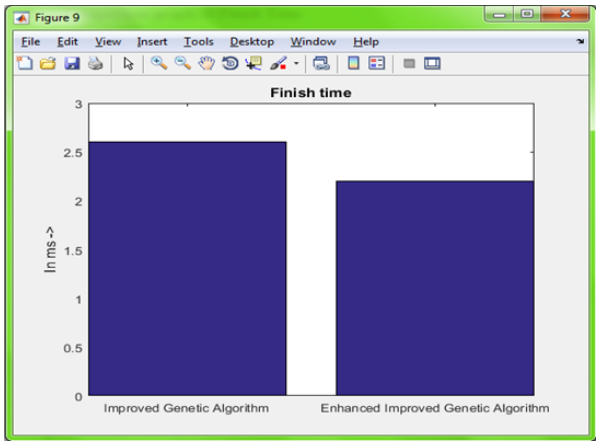

Fig 2: Comparison graph of Finish Time

Figure 2 shows the finish time of the improved genetic and proposed enhanced improved genetic algorithm compared for the performance analysis. The finish time of the enhanced improved genetic algorithm is less as compared to improved genetic algorithm.

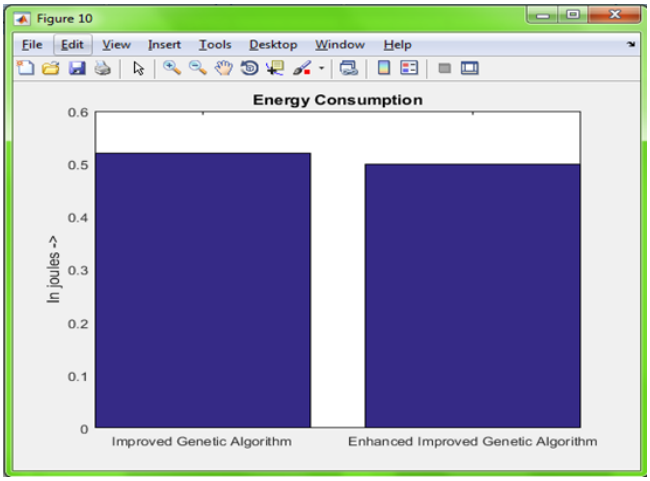

Fig 3: Comparison graph of Energy Consumption

Figure 3 shows the energy consumption of the improved genetic algorithm and proposed enhanced improved genetic algorithm compared for the performance analysis. The energy consumption of enhanced improved genetic algorithm is less as compared to improved genetic algorithm.

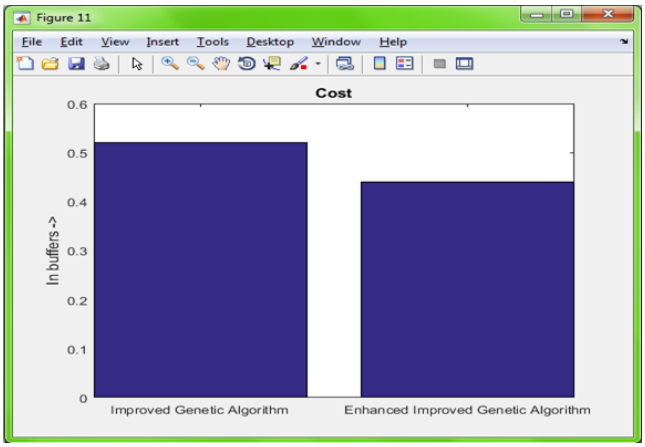

Fig 4: Comparison graph of Cost

Figure 4 shows the cost of the improved genetic algorithm and enhanced proposed improved genetic algorithm compared for the performance analysis. The enhanced improved genetic algorithm has less cost as compared to improved genetic algorithm.

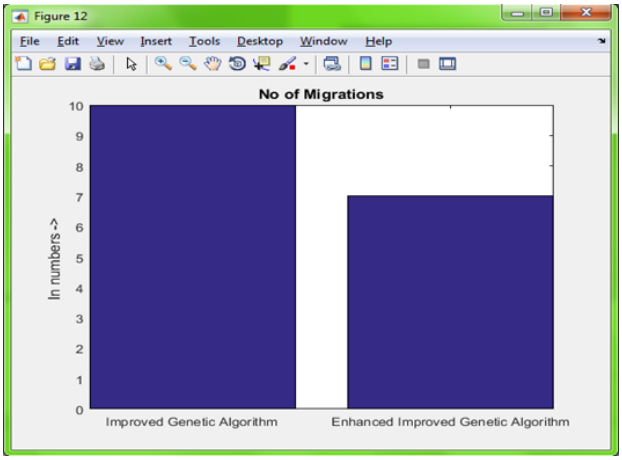

Fig 5: Comparison graph of No of Migrations

Figure 5 shows the number of migrations of improved genetic and proposed enhanced improved genetic algorithm compared for the performance analysis. The number of migrations of enhanced improved genetic algorithm is less as compared to improved genetic algorithm..

Table1: Comparison Analysis

Parameters	Genetic Algorithm	Improved Genetic Algorithm
Response Time	2.7 seconds	2.5 seconds
Finish Time	2.6 seconds	2.3 seconds
Energy Consumption	0.55 joule	0.5 joule
Cost	0.5	0.45
No of Migration	10 migrations	7 migrations

V. CONCLUSION

This work is centered around the load balancing issue being faced in cloud frameworks. In systems, delay can be increased due to improper load balancing. To perform virtual machine migration, previous research has implemented genetic algorithms. It has been seen through this research that the genetic algorithm is quite complex in nature. Therefore, the time of virtual machine migration increases. The aim of this research work is to accomplish virtual machine migration through the implementation of advanced genetic technology. The envisioned algorithm is implemented in the MATLAB software and a number of parametric values are computed for studying the performance of this algorithmic approach. According to the results, the proposed algorithm has better performance than the existing algorithm. In future, this work could be extended by presenting a new security algorithm to isolate virtual channel attack from clouds.

VI. REFERENCES

[1]. Karan D. Patel, Tosal M. Bhalodia, “An Efficient Dynamic Load Balancing Algorithm for Virtual Machine in Cloud Computing”, 2019, International Conference on Intelligent Computing and Control Systems (ICCS)

[2]. Guilin Shao, Jiming Chen, “A Load Balancing Strategy Based on Data Correlation in Cloud Computing”, 2016 IEEE/ACM 9th International Conference on Utility and Cloud Computing (UCC)

[3]. Pradeep Kumar Tiwari, Sandeep Joshi, “Dynamic weighted virtual machine live migration mechanism to manages load balancing in cloud computing”, 2016, IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)

[4]. Narayan Joshi, Ketan Kotecha, D B Choksi, Sharnil Pandya, “Implementation of Novel Load Balancing Technique in Cloud Computing Environment”, 2018,

International Conference on Computer Communication and Informatics (ICCCI)

[5]. P. Geetha, C.R. Rene Robin, "A comparative-study of load-cloud balancing algorithms in cloud environments", 2017, International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)

[6]. T. Deepa, DhanarajCheelu, "A comparative study of static and dynamic load balancing algorithms in cloud computing", 2017, International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)

[7]. Hussain A Makasarwala, PrasunHazari, "Using genetic algorithm for load balancing in cloud computing", 2016, 8th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)

[8]. Dalia Abdulkareem Shafiq, Noor Zaman Jhanjhi, Azween Abdullah, Mohammed A. Alzain, "A Load Balancing Algorithm for the Data Centres to Optimize Cloud Computing Applications", 2021, IEEE Access

[9]. Sreelakshmi, S. Sindhu, "Multi-Objective PSO Based Task Scheduling - A Load Balancing Approach in Cloud", 2019, 1st International Conference on Innovations in Information and Communication Technology (ICIICT)

[10]. Ronak Agarwal, Neeraj Baghel, Mohd. Aamir Khan, "Load Balancing in Cloud Computing using Mutation Based Particle Swarm Optimization", 2020, International Conference on Contemporary Computing and Applications (IC3A)

[11]. Vishalika, Deepti Malhotra, "LD_ASG: Load Balancing Algorithm in Cloud Computing", 2018, Fifth International Conference on Parallel, Distributed and Grid Computing (PDGC)

[12]. Dr. Mani goyal,Avinash Sharma" Enhancing Hybrid Encryption Techniques for Secured Data Processing for Small Medium Enterprises in cloud", 2022, IEEE International Conference on Technology, Research, and Innovation for Betterment of Society (TRIBES) 10.1109/TRIBES52498.2021.9751621

[13]. Dr.Mani goyal Dr.Avinash Sharma" A Hybrid Encryption Model to Lower the Complexity of Securing the Data in Cloud",2020,Journal of Computational and Theoretical Nanoscience(JCTN). https://doi.org/10.1166/jctn.2020.8963

[14]. Dr.Mani goyal,Avinash Sharma" a mobile-cloud framework with active monitoring on cluster of cloud service providers",2019, 2nd International conference on innovative computing and communication,ICICC 2019.

[15]. Dr.Mani goyal,Avinash Sharma", Framework for Integrated Communication of Mobile and Cloud service provider via Cloud cluster with Homomorphism Encryption Technique""6 sep 2021" 4th IEEE International Conference on Systems, Computation, Automation and Networking (ICSCAN 2021) https://ieeexplore.ieee.org/abstract/document/9673454

[16]. Dr.mani goyal ,Avinash Sharma" Implementation and Analysis of various Encryption Techniques with Blowfish on various Data Files",2021"" International Conference on Technological Advancements and Innovations (ICTAI - 2021)".