IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-7, Issue-5, May 2019 E-ISSN: 2347-2693

Dynamic Fault Tolerance Job Allocation Mechanism to Conserve Resources in Vehicular Cloud

Jaspreet Singh[1*], Kamaljit Kaur[2]

Department of Computer Engineering and Technology
Guru Nanak Dev University, Amritsar, Punjab, India

Corresponding Author: erjaspreetsingh123@gmail.com

DOI: https://doi.org/10.26438/ijcse/v7i5.538547 | Available online at: www.ijcseonline.org

Accepted: 22/May/2019, Published: 31/May/2019

Abstract -Today sensing resources are widely increased in terms of vehicles and it affects the cloud computing systems. This technology is used for predicting traffic and for road safety. These systems usually share resources and collaborate with sensing devices for processing data and propagate results. In this paper we proposed Vehicular cloud based fault tolerance mechanism that considers cost matrix and dynamic fault tolerance. The allocation of resources depends critically on the cost associated with virtual machine. It considers exponential residency of VC and execution time along with bandwidth utilization. Bandwidth consumption and cost of execution is reduced greatly by the effect of proposed mechanism.

Keywords**:** Vehicular Cloud, Cloud computing, Fault tolerance, Resource scheduling.

I. INTRODUCTION

Today the promising area is utilizing vehicular network along with mobile cloud computing that is known as vehicular cloud computing.[1] It is an integrated platform that shrinks the spectrum of services and provides data dissemination. This technology provides links between infrastructures to vehicles and gives better services to the connected clients.

Cloud Computing offers access to resources that are shared which includes storage space, computation power, network, applications and services on demand basis to the users over the internet. [2]The user no longer need to worry about the initial investments on the resources since cloud computing provides an approach for leveraging computing resources with same ease as utilizing common utilities such as natural gas, water , electricity supply on pay per use bases through the concept of utility oriented computing thus ensuring Quality of Service at the same time.

It gives shared services to organization along with local servers and storage resources. [3]The information can be remotely accessed from the cloud. Here everything is dependent upon cloud so all processing is done in cloud that needs commands for taking actions in network. But the network may not have enough bandwidth so it utilizes cloud computing services. In this the vehicles data is uploaded over the cloud and then the information is maintained. After that according to this information that is saved in data repository over the cloud the ID for each vehicle is generated and it is saved on the cloud. This ID is further utilized for accessing the vehicles and assigning the resources to the vehicles.

[4]The task of overseeing networking resources turns out to be testing when utilizing diverse framework, networking advances and application that are distributed. Likewise, as disseminated frameworks increment in size and unpredictability, it winds up important to computerize a portion of the management undertakings. [5]At the point when this circumstance is contrasted and traditional management situations, it is discovered that in such frameworks manager use strategy as the reason for settling on operational management choices .Management framework where strategy server design been created for clog management. The arrangement of approaches and their principles been conveyed in the framework for overseeing and controlling the blockage in network. In this paper we proposed a VANET based fault tolerance mechanism that handles faults efficiently.
Rest of the paper is organized as under: section ii gives the literature survey, section 3rd gives the proposed methodology, section 4th gives the result and performance analysis, section 5th provides conclusion, section 6th gives references used in the proposed work.

II. LITERATURE SURVEY

This section gives the literature of techniques used to provide optimization within vehicular cloud.

[6]The adaptive optimization of the aforementioned scheduling actions has been till now pursued by following three main parallel (e.g., disjoint) research directions, so that the proposed cognitive computing inspired joint approach seems to be somehow new. Specifically, a first research direction focuses on task scheduling algorithms for the minimization of the energy in reconfigurable data centers that serve static clients .

For this purpose, in [7], an energy-efficient greedy type scheduling algorithm is presented. It maps jobs to VMs and, then, maps VMs to Dynamic Voltage Frequency Scaling (DVFS)-enabled servers under upper bounds on the allowed per-job execution time. Interestingly, the resulting scheduler relies on a suitable version of the First-Fit-Decreasing (FFD) heuristic, in order to perform the most energy efficient VM to-server mapping and assignment of the processing rates to the servers. Its energy performance is, indeed, noticeable. However, we note that: (i) the application scenario of[8] does not consider, by design, mobile clients; and, (ii) the admission control of the input workload is not performed and the bandwidths of the intra-datacenter links are assumed fixed.

Energy-efficient joint reconfiguration of the overall communication-plus-computing resources in virtualized data centers is the topic of [9], where an approach based on the stochastic nonlinear optimization is pursued. Although the resulting scheduler is adaptive and guarantees bounded per-job execution times, we point out that: (i) the application scenario of does not consider mobile clients; and, (ii) resource consolidation and admission control of the input traffic are not performed.

Energy-saving dynamic provisioning of the computing resources in virtualized green data centers is the topic of [10]. Specifically, in order to avoid the prediction of future input traffic, the authors of [11] resort to a Lyapunov-based technique that dynamically scales up/down the sizes of the VM’s tasks and the corresponding processing rates by exploiting the available queue information. Although the pursued approach does not suffer from prediction errors, it relies on an inherent execution delay-vs.-utility tradeoff, that does not allow us to account for hard deadline on the execution times.

A second research direction deals with the energy efficient traffic offloading from VCs to serving RSUs by exploiting the underlying Vehicular-to-Infrastructure (V2I) wireless connections [11], [12].

In this framework, the focus of [13] is on the optimized task and processing rate mapping of an assigned Task Interaction Graph (TIG) to a computing graph composed by networked reconfigurable computing nodes. As in our approach, hard constraints on the overall execution times are considered by [13]. However, unlike our approach, we point out that: (i) the focus of [14] is on the traffic offloading from mobile devices to remote clouds, so that the resulting task scheduler does not support, by design, data dissemination and content delivery services; (ii) the joint task and computing rate mapping afforded in [15] is, by design, static; and, (iii) the scheduler in [16] does not perform real-time reconfiguration and/or consolidation of the computing resources hosted by the serving cloud.

The authors of [17] develop a distributed scheduler for performing mobile-to-access point traffic offloading. The scheduler enforces cooperation among the mobile devices, in order to minimize the average energy consumption of the mobile devices under per-device hard upper bounds on the volumes of the traffic to be offloaded to the remote cloud. For this purpose, in [18], it is assumed that mobile devices can execute tasks locally, offload tasks to other cooperative mobile devices or to the remote cloud, in accordance with the decision taken by the proposed scheduler. However, we point out that: (i) the work in [13] does not consider a mobile scenario; (ii) the scheduler in [13] does not enforce per-client QoS guarantees on the maximum execution-plus communication delay and/or the minimum processing rate; and, (iii) the energy wasted by the remote cloud for processing the offloaded traffic is not included in the objective function of [13].

A third research direction focuses on the energy consumption of hand-held wireless devices when traffic offloading over 3G/4G/WiFi connections is performed [14], [15], [16].

III. PROPOSED SYSTEM

Vanet based fault tolerance in cloud computing provides reliable result by reducing the congestion and assigning load to the virtual machines having least load. The starvation problem thus being eliminated using this system. The mechanism also employ cost matrix that includes cost that is being encountered when job is allotted. The architecture of the proposed mechanism is given in figure 1.

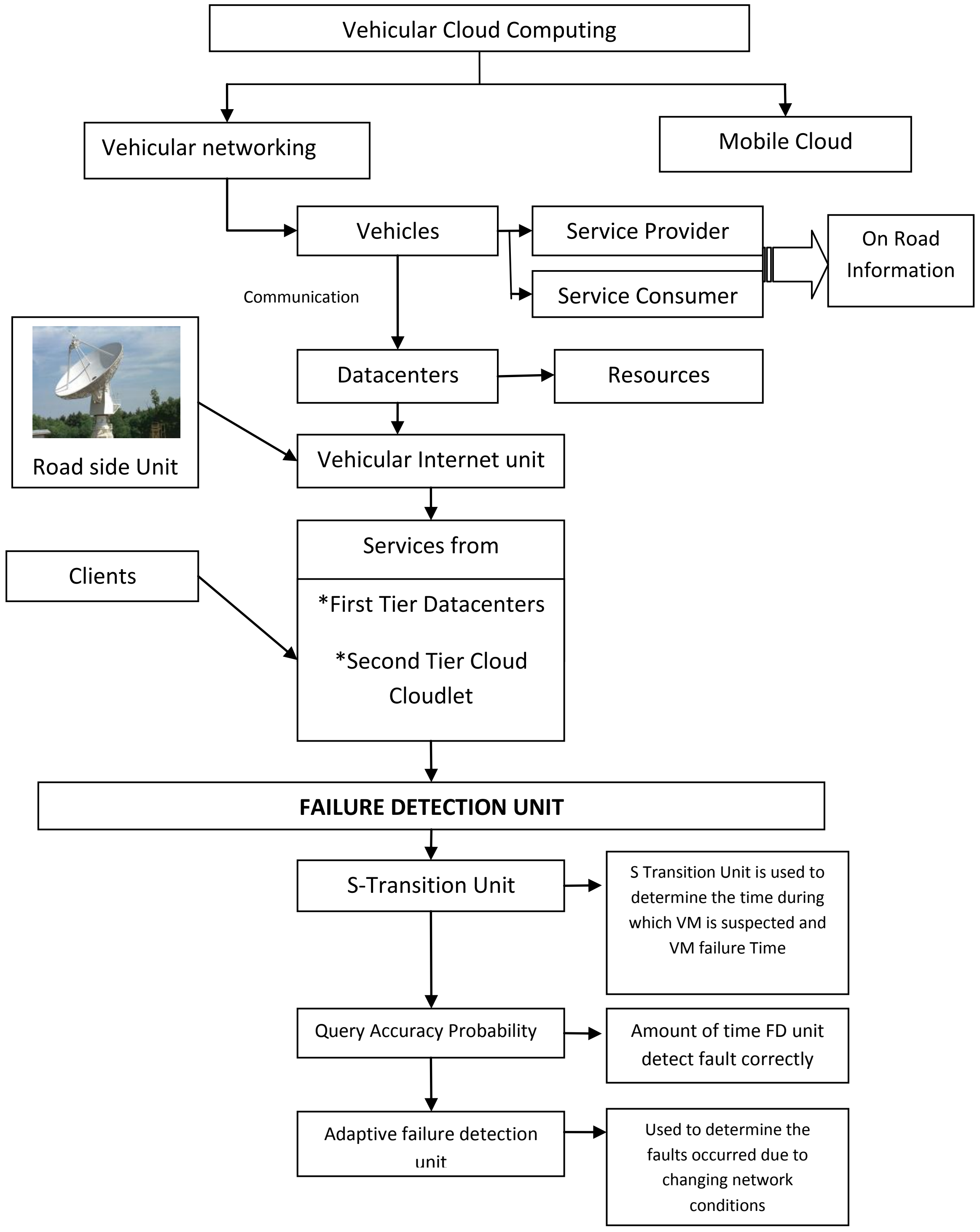
Vehicular Cloud Computing
Vehicular networking
Mobile Cloud
Vehicles
Service Provider
Service Consumer
On Road Information
Communication
Road side Unit
Datacenters
Resources
Vehicular Internet unit
Clients
Services from
*First Tier Datacenters
*Second Tier Cloud Cloudlet
FAILURE DETECTION UNIT
S-Transition Unit
S Transition Unit is used to determine the time during which VM is suspected and VM failure Time
Query Accuracy Probability
Amount of time FD unit detect fault correctly
Adaptive failure detection unit
Used to determine the faults occurred due to changing network conditions

Fig 1: Architecture of the proposed system

This figure describes the structure of VANET introduction in cloud. The factors affecting the job execution in cloud using the application of VANET is given through this figure. The figure elaborate terminology and fault detection mechanism employed within the VANET based cloud.

A. Description of the proposed system

The description of the proposed system as given below:

- **Task Handler**

Task handler is situated in Cloudlet side which first receives the task request from the mobile device. Task handler mainly analyzes the task and makes a weight for the computational benefit from the cloud. This task handler initially assesses the whole task and the priority and importance. For the further analysis then it send to the aggregator and profile section.

- **Aggregator**

The aggregator mainly aggregates all the components of the task and reorganizes and rearranges it according to the sequence of the application. It basically incorporates the different components of the task and reshuffles and reorganizes the random part to make it in an orderly meaningful manner.

- **Profiler**

Profiler mainly responsible for analyze the applications and its different components. It describes the need of demand for computation units need to complete the task execution. In addition it can make the all components of profile to identify and reorder the task and calculate the whole necessary resources needed to finish the work.

- **Analyzer and Optimizer**

After profiling, tasks are handed to the analyzer and optimizer section for further analysis based on the profiler data. By using optimization techniques, analyzer and optimizer decide to further processing of the request tasks which should be optimally done by the available resources for the cloudlet and mobile devices.

- **Scheduler**

Scheduler makes the scheduling for complete the task. It assigns the priority which devices the task will reschedule and overall available resources for task scheduling. When task finish then it update accordingly and make the resource free for the other sequential important task for completion.

- **Partition/Decision Engine**

The decision engine makes the decision of application execution. It could be the in to cloudlet or to the mobile devices other than the sending devices. This section makes decision after 541igratory the resources associated with the mobile devices resources. We employ several decision making algorithm in this stage to get the optimized and enhanced performance by make the decision that should be the optimized one.

- **Mobility Manager**

Mobility manager hold the status of all the mobile devices connected to the cloudlet in the proximity of cloudlet. It can be done by storing all the Wi-Fi signal strength registered with the Wi-Fi zone and their signal strength confirm us the proximity of the all the nearest devices. Among them, from the client profiler, we can have the resources latest update and the information who are waiting for the cloudlet services by the receiving the results from the cloudlet. Mobility is an important aspect which we also consider in our framework. From the mobility pattern analysis, when decision engine make decision which mobile device, it is going to offloaded task to the client devices can help the predicted mobile device.

- **Resource Manager**

Resources indexing are keeping as a database to the cloudlet. All the available and presently used resources are keep track by the resource manager. Therefore resource manager is responsible for keep all the updated data and information in the cloudlet which is very important for the decision engine to estimate the available resources from the cloudlet. Hence, every new device joining in the network should register their resources by the profiler to the cloudlet and again, when any device leave the network, instantly the resource manager remove the device and it's available resources from the database.

- **Migrator**

Cloudlet use Migrator to transfer the portion of data or code segment or process to migrate to the surrounded mobile devices. Obviously, decision engine using our algorithms make decision to choose the best case mobile device. After that for sending the code, 541igratory is responsible of sending to the mobile device. Migrator receives the result from the mobile device to process it again by sending to the aggregator and other further steps.

- **Client Devices Resources**

The resources which are embedded with the client mobile devices are mentioned or marked here as user device resources. It could be CPU, memory or even storage. Several cases it could be the installed software that cannot be processed by the cloudlet or clod let has not installed with the software facility. We can have huge sensors now a day that could be great resources as well such as GPS, camera, thermometer, location apps, and embedded other latest sensors.

- **Cloudlet Resources**

Resources which are belongs to cloudlet are normally considered as CPU, memory, and hard disks. In this experiment and our frame work, mainly we consider the compute-intensive applications, hence we identified the CPU is the main resources. As, we proof in chapter 1 that certain cases, this resources are not adequate and we call it some point it as a resource scarcity which degrade the overall cloudlet performance negatively. We aim to consider the problem and propose the solution to shift some application or process from the cloudlet to the nearest mobile devices that they can act as collaborative resources for the cloudlet. Since when the loads are reduced from the cloudlet, ultimately the performance has been increased and the results we get from the nearest mobile devices as working as a resource node or hub eventually make the whole process faster and make the cloudlet to increase the performance and can get rid of resource scarcity.

The flow of proposed system is given in figure 2

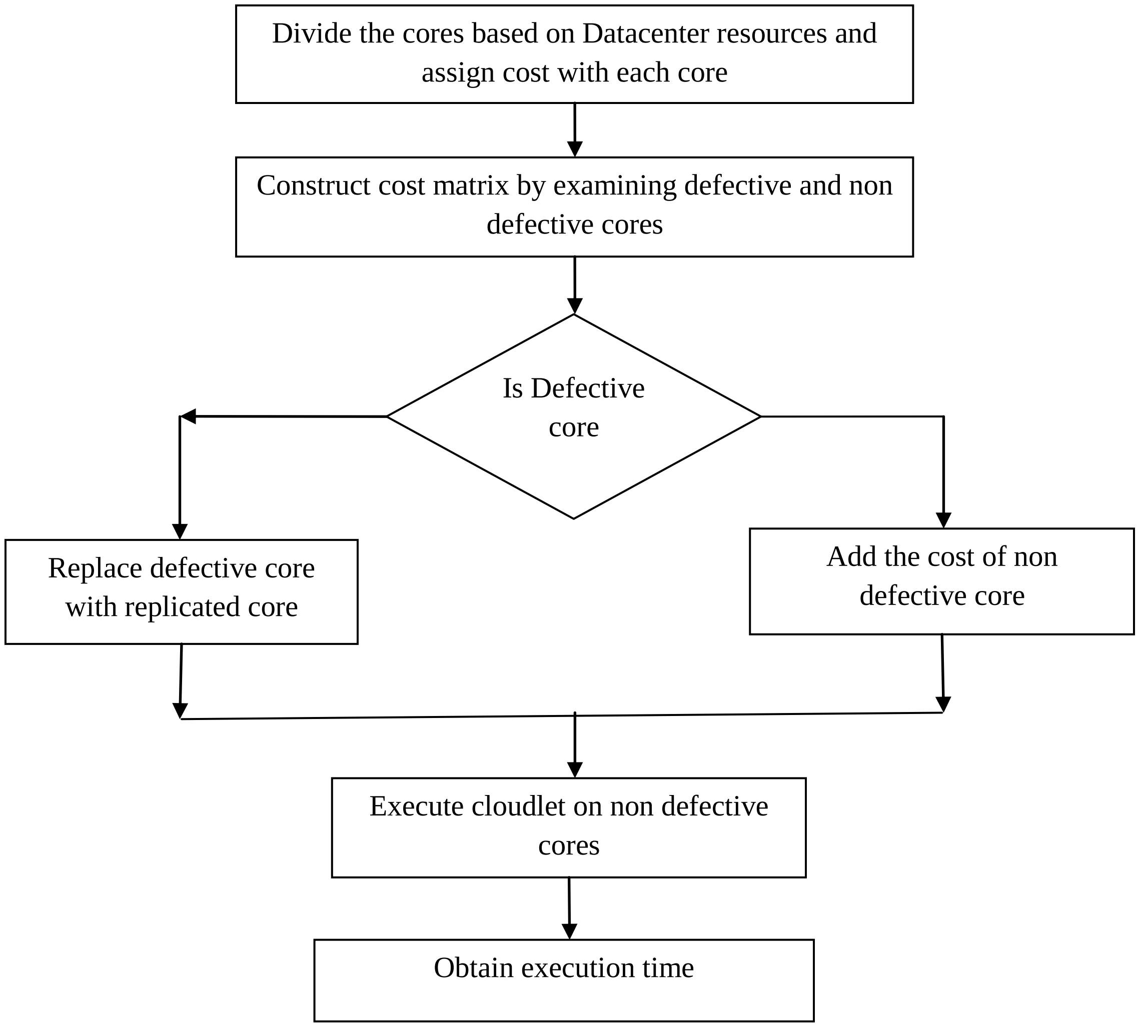
Divide the cores based on Datacenter resources and assign cost with each core
Construct cost matrix by examining defective and non defective cores
Is Defective core
Replace defective core with replicated core
Add the cost of non defective core
Execute cloudlet on non defective cores
Obtain execution time

Fig 2: Flow of proposed system

The fault tolerance unit is the main objective of this literature. This unit detects the faults from the core and performs the migration. The migration performed through this system decrease the overall cost associated with the transmission.

IV. RESULTS AND DISCUSSION

The result obtained from the proposed system shows optimization. The result obtained interms of distinct parameters. The parameters used in the proposed system includes average cost, execution time and fault tolerance rate.
The cost encountered in terms of overhead is presented through Table below :

Table 1: Cost matrix comparison with cloudlets

Cloudlets	Without Cost Matrix	With Cost Matrix	With Cost matrix and dynamic load
100 Cloudlets	4.3	4.1	3.2
200 Cloudlets	5.5	5.4	4.9
300 Cloudlets	6.3	6.1	5.3

400 Cloudlets	7.5	6.8	6.5

$$\text{Cost} = VM\_cost * Total\ cloudlets\ executed$$

The cost encountered in terms of overhead is presented through figure 3

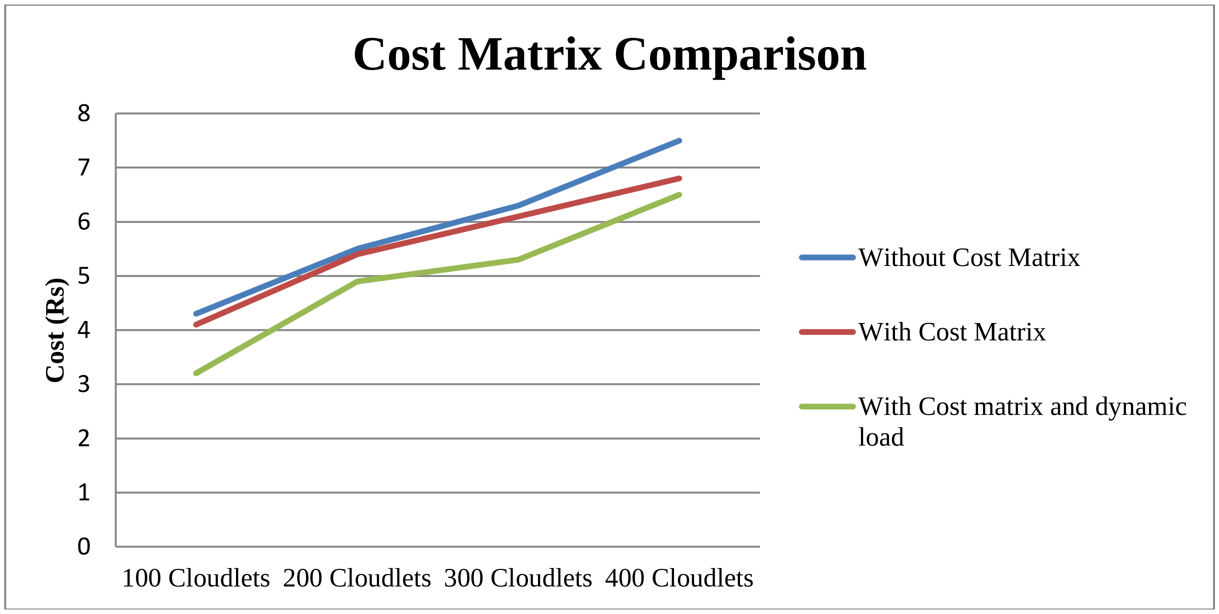

Fig 3: Cost Comparison with existing and proposed system.

The cost or overhead increases as number of cloudlet increases. The cost matrix used shows deviation and falls very close to static allocation mechanism at least load. The load when increases cost vary exponentially. The execution time indicating difference between start and end time. The table below shows the comparison between execution time.

Table 2: Execution time comparison for cloudlets

Cloudlets	Without cost matrix	With Cost matrix	With Cost matrix and dynamic load
100 Cloudlets	15.2	14.5	10.9
200 Cloudlets	16.7	14.9	11.5
300 cloudlets	18.2	15.3	12.1
400 Cloudlets	19.7	15.7	12.7

$$Execution\ time = finis\square\_time - Start\_time$$

The execution time comparison is shown with the help of figure 4.

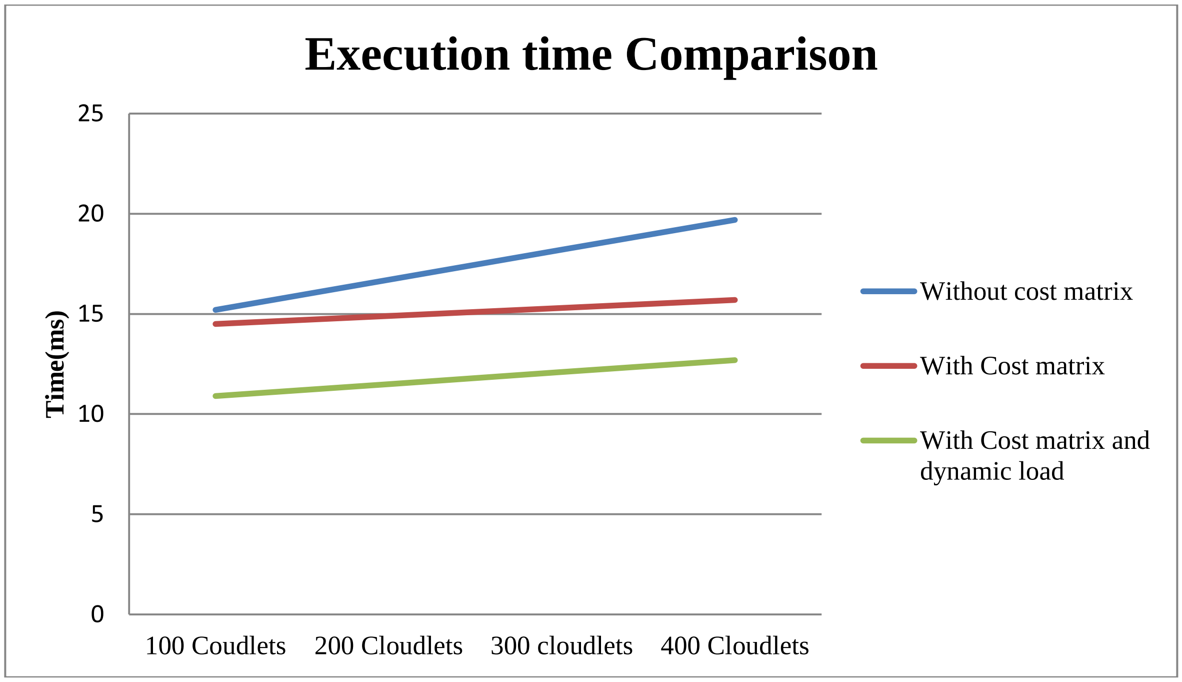

Fig 4: Execution time comparison

The execution time decreases as the machine selected for load allocation is optimal. The load decreases with the application of proposed system. The execution time is estimated using the standard command system.currenttimemillis() of netbeans.
The fault tolerance rate increases as the optimal virtual machine is selected for load allocation. The load allocation and fault tolerance rate is given through the figure 5.

$$Fault\ tolerance\ rate = no.\_of\_faults\ /\ Execution\_time$$

Table 3: Fault Tolerance rate comparison

Cloudlets	without cost matrix	with cost matrix	with cost matrix and dynamic load
100 cloudlet	0.3	0.4	0.5
200 cloudlet	0.29	0.38	0.498
300 cloudlet	0.26	0.37	0.47
400 cloudlet	0.24	0.3533333	0.46

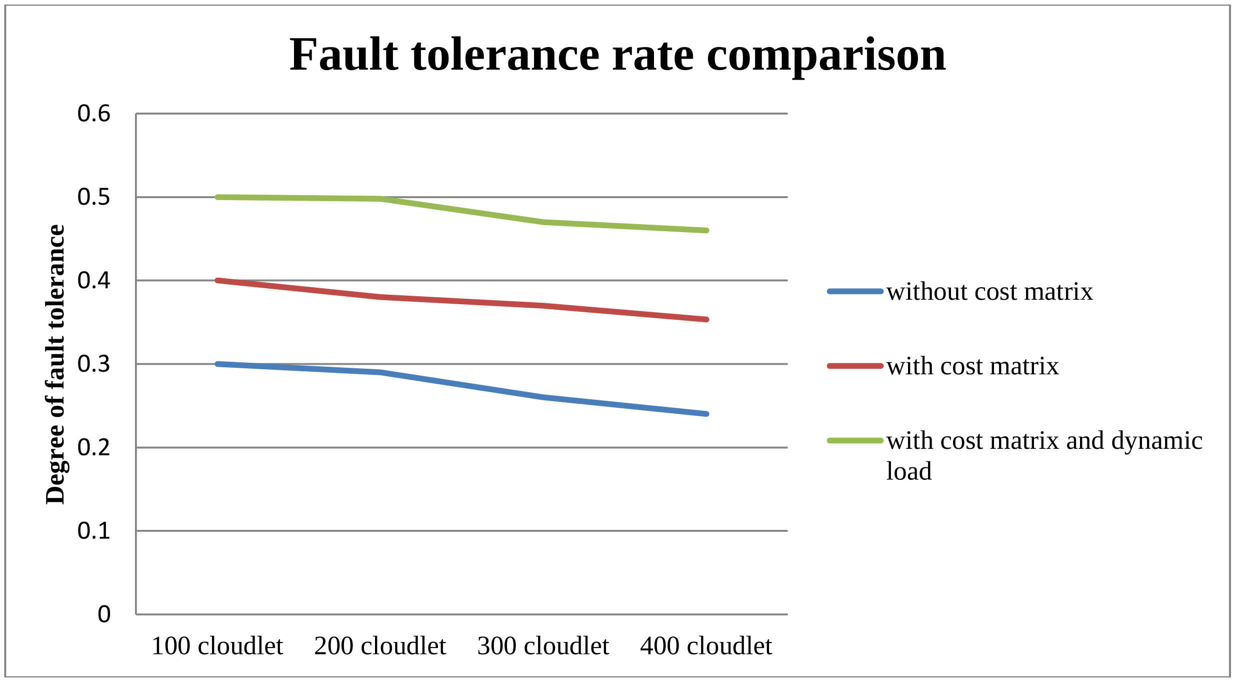

Fig 5: Degree of fault tolerance

The degree of fault tolerance is the criteria through which worth of study can be checked. The fault tolerance rate is significantly higher in case of proposed system as compare to existing approaches. The dynamic load variation is the primary cause of improvement of fault tolerance rate. The static cost matrix mechanism also shows improvement however since same machine is selected again and again hence congestion causes the problem. Dynamic cost matrix formation however improves the overall parametric results.

MTTF (mean time to failure): It is the basic unit of measuring reliability of system. The total time used by an item during start of execution till its failure is known as MTTF. In the other words it represents the lifetime of a system. In the proposed system mean time to failure increased and the reliability of system enhanced significantly as compared to existing system. The figure below shows the MTTF of proposed and existing system.

$$MTTF = total\ \square ours\ of\ time\ /\ total\ number\ of\ units$$

Table 4: Mean Time to Failure rate comparison

Cloudlets	without cost matrix	with cost matrix	with cost matrix and dynamic load
100 cloudlet	0.56	0.54	0.66
200 cloudlet	0.63	0.697	0.72
300 cloudlet	0.624	0.68	0.76
400 cloudlet	0.73	0.75	0.86

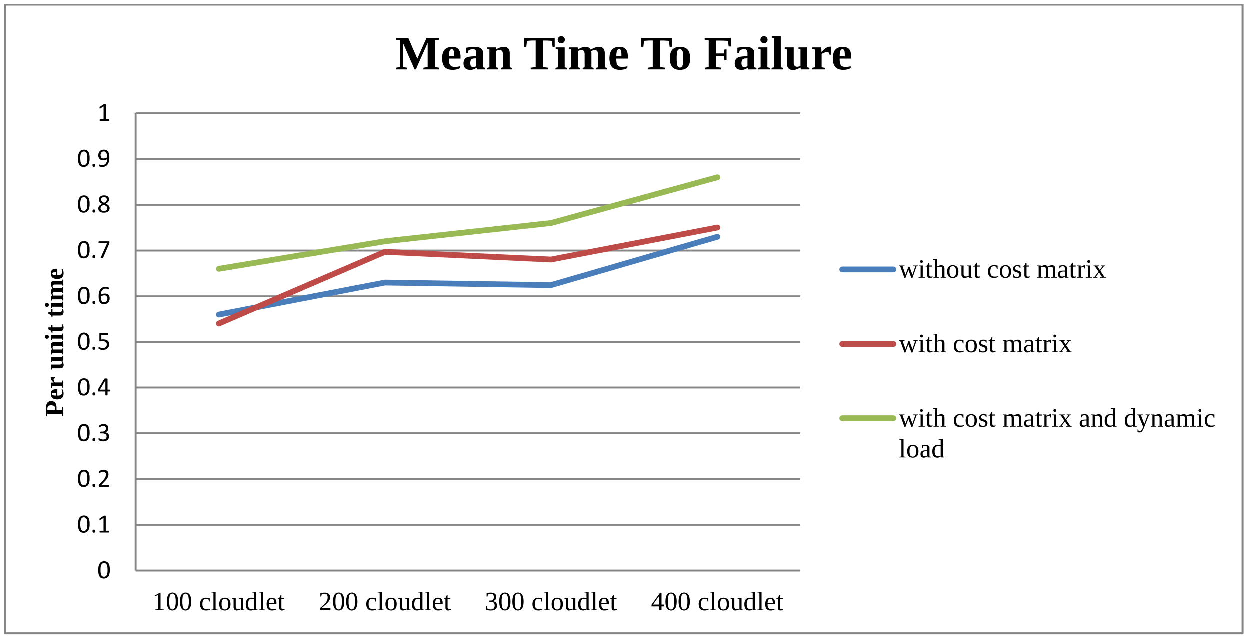

Fig 6: Mean time to failure

V. CONCLUSION & FUTURE SCOPE

The vehicular cloud used for fault tolerance provides new methodology for improving performance of cloud. Virtual machines are mobile in this case. The load allocation considers the cost associated with the virtual machine. This cost varies as the virtual machines shows deviation in terms of load. The virtual machine is discarded from cost matrix having higher load. This can cause steep hike in cost but congestion decreases. Thus parametric comparison shows improvement and cloudlets are executed with greater rates as compared to existing system.

Further it must be enhanced in the field of IOT to conserve the energy of sensor network. In future the vehicular cloud based framework can be designed for intelligent transport system.

REFERENCES

[1] R. K. Naha, S. Garg, D. Georgakopoulos, P. P. Jayaraman, L. Gao, Y. Xiang, and R. Ranjan, "Fog computing: Survey of trends, architectures, requirements, and research directions," IEEE Access, vol. 6, no. c, pp. 47980–48009, 2018.
[2] B. Brik, N. Lagraa, N. Tamani, and A. Lakas, "Renting out Cloud Services in Mobile Vehicular," Res. Gate, no. December, 2018.
[3] M. R. Jabbarpour, A. Marefat, A. Jalooli, and H. Zarrabi, "Correction to : Cloud-based vehicular networks : a taxonomy , survey , and conceptual hybrid architecture Could-based vehicular networks : a taxonomy , survey , and conceptual hybrid architecture," Wirel. Networks, no. November, 2017.
[4] R. Yu, X. Huang, J. Kang, J. Ding, S. Maharjan, S. Gjessing, and Y. Zhang, "Cooperative resource management in cloud-enabled vehicular networks," IEEE Trans. Ind. Electron., vol. 62, no. 12, pp. 7938–7951, 2015.
[5] T. Mori, Y. Utsunomiya, X. Tian, and T. Okuda, "Queueing theoretic approach to job assignment strategy considering various inter-Arrival of job in fog computing," 19th Asia-Pacific Netw. Oper. Manag. Symp. Manag. a World Things, APNOMS 2017, pp. 151–156, 2017.
[6] K. Zheng, H. Meng, P. Chatzimisios, L. Lei, and X. Shen, "An SMDP-Based Resource Allocation in Vehicular Cloud Computing Systems," IEEE Trans. Ind. Electron., vol. 62, no. 12, pp. 7920–7928, 2015.
[7] K. Zhang, Y. Mao, S. Leng, Q. Zhao, L. Li, and X. Peng, "Energy-efficient Offloading for Mobile Edge Computing in 5G Heterogeneous Networks," IEEE Access, vol. 3536, no. c, pp. 1–10, 2016.
[8] W. Zhang, Z. Zhang, and H. Chao, "Cooperative Fog Computing for Dealing with Big Data in the Internet of Vehicles : Architecture and Hierarchical Resource Management," IEEE Access, no. December, pp. 60–67, 2017.
[9] J. Fan, R. Li, and X. Zhang, "Research on fault tolerance strategy based on two level checkpoint server in autonomous vehicular cloud," Proc. 2017 IEEE 7th Int. Conf. Electron. Inf. Emerg. Commun. ICEIEC 2017, no. 61363079, pp. 381–384, 2017.

[10] Y. Sharma, B. Javadi, W. Si, and D. Sun, "Reliability and energy efficiency in cloud computing systems: Survey and taxonomy," J. Netw. Comput. Appl., vol. 74, pp. 66–85, 2016.
[11] H. S. Y. Lin, "EAFR: An Energy-Efficient Adaptive File Replication System in Data-Intensive Clusters," IEEE Trans. Parallel Distrib. Syst., pp. 1017–1030, 2017.
[12] B. Shrimali and H. Patel, "Performance Based Energy Efficient Techniques For VM Allocation In Cloud Environment," IEEE Access, pp. 477–486, 2017.
[13] H. . Z. D. . Zhao B.a Aydin, "Reliability-Aware dynamic voltage scaling for energy-constrained real-time embedded systems," 26th IEEE Int. Conf. Comput. Des. 2008, ICCD, vol. 546244, pp. 633–639, 2008.
[14] H. M.-R. Mahdi Ghamkhari, "Energy and Performance Management of Green Data Centers: A Profit Maximization Approach," IEEE Trans. Smart Grid, pp. 1017–1025, 2017.
[15] M. Salehi, M. K. Tavana, S. Rehman, S. Member, M. Shafique, and A. Ejlali, "Two-State Checkpointing for Energy-Efficient Fault Tolerance in Hard Real-Time Systems," pp. 1–12, 2016.
[16] S. Ben Alla and A. Ezzati, "Hierarchical adaptive balanced energy efficient routing protocol (HABRP) for heterogeneous wireless sensor networks," Ieee, 2011.
[17] P. Handa, B. Singh Sohi, and N. Kumar, "Energy efficient hybrid routing protocol for underwater acoustic sensor network," 2016 Int. Conf. Electr. Electron. Optim. Tech., pp. 2573–2578, 2016.
[18] B. Mills, T. Znati, R. Melhem, K. B. Ferreira, and R. E. Grant, "Energy consumption of resilience mechanisms in large scale systems," Proc. - 2014 22nd Euromicro Int. Conf. Parallel, Distrib. Network-Based Process. PDP 2014, pp. 528–535, 2014.

AUTHORS PROFILE

Jaspreet Singh did Bachelor of Technology in Computer Science from Amritsar College of Engineering And Technology , Amritsar in year 2017. He is currently pursuing Masters of Technology in Computer Science from Guru Nanak Dev University, Amritsar and currently working as Research Scholar in Department of Computer Engineering and Technology. His main area of research focuses on Fault Tolerance in Cloud Computing.

Ms. Kamaljit Kaur did Bachelor of Technology and Master of Technology in Computer Science from Guru Nanak Dev University. She is the Gold Medalist in Masters of Technology. She is currently pursuing Ph. D. and working as Assistant Professor in Computer Engineering and Technology Department. She has published more than 15 research papers in UGC Approved Journals. Her main research area focuses on Cloud Computing, Parallel Computing and Fault Tolerance.