Shadow Replication using Dynamic Core Allocation for Application Fault Tolerance in Virtual Environment

Navpreet Kaur Gill[#], Kamaljit Kaur[*]

[#] *Department of Computer Engineering & Technology, Guru Nanak Dev University, Amritsar, Punjab, India*

***Abstract*— As the interest for cloud computing keeps on increasing, cloud specialist organizations are confronted with the overwhelming test to meet the targeted SLA agreements, as far as dependability and convenient execution is concerned, while accomplishing cost and energy efficiency. This paper proposes Shadow Replication, a novel fault-tolerance mechanism for cloud computing, which flawlessly addresses fault at scale, while limiting energy utilization and lessening its effect on cost. Energy conservation is achieved by creating dynamic cores rather than static cores. Cores are created by the application of cloudlets. In other words proportionate cores are created. Core failure metrics are considered to be memory capacity, energy and power consumption. In case any of the parameter exceeded threshold value, core is supposed to be faulted and progress is maintained within shadow which is maintained 1 per VM. Progress of deteriorated core is shifted to next core within same VM. In case all the cores within the VM deteriorate, VM migration is performed. Results obtained by allocating cores dynamically reduce energy consumption, latency, cost and maximize fault tolerance rate because of reduced VM migration overhead.**

***Keywords*— Fault Tolerance, Energy Efficiency, Shadow Replication, Dynamic cores, Core Failure, Application level Failure.**

I. INTRODUCTION

Cloud Computing has been developed as an attractive option for process and information escalated applications, because of its low costs, on demand provisioning of services and diminished cost of keeping up interior IT framework [1].Cloud computing will proceed to develop and sustain in the market because of the immense advantages it offers to established as well as newly set up IT firms. Late reviews anticipate yearly development rate of 17.7 percent by 2016, making cloud computing the quickest developing area in the product business [2]. Along with the benefits of cloud comes the constant burden of meeting elevated requirements of dependability, viability, and expandability [3]. These frameworks ought to regularly meet hard due dates and the inability to meet the due dates could have unfortunate outcomes. To avoid these circumstances, data centres constantly expand by adding more servers, networking devices and storage components so that high rising demands could be met within fraction of seconds, giving clients' high quality experience and enhancing performance to the fullest. However, as the size of data centres increase, two major issues are faced namely failures and high energy utilization.

As the data centres scales, the probability of failure occurrences also increases. With the addition of resources comes the constant threat of their failure. It is because of this reason that it is anticipated that in future extreme scale systems, the failure will occur in the order of hours or even minutes. If the failures occur at such a pace, the performance and user experience will be severely degraded because of increasing downtime. Keeping in mind the dependence on cloud computing framework, appropriate fault handling mechanisms need to be designed.

Moreover, as the data centres expand and more servers, networking and storage components are added, the energy required to run these resources also increases. The storage devices account for 27% of total power expenditure in a classic data canter [4]. According to [5], IT equipments consume 40% of total energy which is shared between computing servers and network equipments. Given such high energy utilization patterns by these components, it can only be anticipated that these figures will increase will time given the dependence on cloud computing technology and constant growth in this area. In 2005, the power consumption of a single supercomputer was 3.2 Megawatt, which increased two folds in just five years mounting to 17.8 Megawatt which is a tremendous increase. So the need of the hour is energy conservation.

The need to tackle two pressing issues that is failures and energy utilization, calls for the need of an efficient fault handling mechanism that is energy conserving but at the same time highly resistant against failures. The most popularly used fault tolerance techniques in cloud are checkpointing and replication. But none is energy conserving in nature. It is predicted that in future as more and more components will be added and failures will become a norm rather than an exception, checkpointing performance will degrade because of having to do excessive checkpointing to deal with rising level of failures. The time to checkpoint and restart from the saved state will eventually approach system's mean time between failures, degrading performance. Similarly replication requires hardware redundancy to maintain duplicates of the same task and run them in parallel so that even if failure occurs atleast one instance successfully completes the work. But this technique consumes great deal of resources. To carry out process replication, atleast doubles the amount of compute nodes are required. Although massive research is carried out to make these two approaches energy efficient also discussed in this paper, but still a standalone approach is required that can manage fault and at the same time is energy efficient in nature.

To address this issue, shadow replication was proposed that aimed at managing replicas of the process and running them in parallel with the main process but at a reduced rate. Reducing the speed at which replicas run brings huge energy savings compared to checkpointing and traditional replication approach.

This paper introduces Shadow replication using dynamic core allocation for application fault tolerance in virtual environment. As it is known that virtualization provides immense benefits, be it judicious use of resources, energy conservation or providing high resilience to failures. Combining this approach with shadow replication further enhances it. This approach is further extended to dynamic core allocation which brings benefits in terms of energy savings, latency, cost and fault tolerance rate when compared to static approach. The contribution of this paper is:

1. To design a scheduling approach that assigns tasks to the virtual machines in an energy efficient manner.
2. To tolerate core level and application level failure in an efficient manner.
3. To design an energy aware approach whereby power limits are defined and it is ensured that the virtual machine works under the assigned power ranges.
4. To design energy efficient virtual migration approach.
5. To design dynamic core allocation policy and check for improved core static policy in terms of energy consumption, latency, cost and fault tolerance rate attainable.

II. LITERATURE SURVEY

Fault Tolerance and Energy Efficiency have remained two immensely researched topics of all decades. However, collaborating these two areas and studying them as one is a more recent trend. Great deal of work is done on making fault tolerance approaches energy friendly. Checkpointing and Replication are two most widely used techniques in cloud computing infrastructure. The two have been studied length and breadth in an effort to make them immensely reliable and energy effective.

In case of energy efficient checkpointing, [6],[7],[8] were the first ones to give energy estimation framework for checkpointing protocols without even pre-running the applications which helped choosing the most energy conserving protocol. Apart from estimation, [9],[10] monitored energy footprints of checkpoint/rollback based operations by chunking the process of checkpointing into subparts such as saving checkpoints and restarting from the saved checkpoint. After granulizing, energy consumption of each subpart is measured and added up to get the energy consumption of the entire process. Furthermore, energy optimization of checkpointing operations is studies by using DVFS approach [11], [12], preventing useless checkpoints [13], compressing the checkpoints to be stored in the memory [9], optimizing checkpoint schedules [14] and checkpoint chunks [15].

In case of energy efficient Replication, massive study is conducted across three horizons of cloud services namely storage, networking and computing. In case of storage level replication, [16],[17],[18],[19] have studied how to effectively manage data replicas to be stored by determining exactly what data to replicate, how many minimum copies to create and where to store these copies such that the total energy consumption drops. Furthermore, the storage disks can be switched to standalone state in case of light loads to conserve additional energy [20],[21],[22]. Moving ahead to network level, work is done on reducing delays to save energy by replicating data close to the site of usage [23][24]. At computational level, servers are zoned into hot servers with CPU utilization greater than 0 and cold servers with CPU utilization greater than 0. Popularly used data is maintained in hot servers whereas the cold data is kept in the cold servers to save energy [25],[26]. Furthermore,[27] proposed to maintain the data replicas on switched off servers on remaining active servers in balanced manner so that the access to data on powered down servers is not lost at any point of time.

Apart from these basic strategies to save energy in replication, shadow replication is developed as a standalone energy conserving replication approach [28][29][30][31][32]. In case of shadow replication a suite of shadows or in other words replicas are attached to each primary task. While the primary runs at full speed, the shadows execute at a lessened pace to conserve energy. The failure of primary task makes one of the shadows active by enhancing it pace so as to catch up with the work of failed primary and complete it successfully.

The mechanism used to control the execution pace of the shadows is either via the use of dynamic voltage and frequency scaling approach [28][29][30][31][32] or by collocating multiple tasks in the same core and using time sharing to attain the required execution pace while running the cores are maximum frequency[33][34]. The exiting work on shadow replication is summarized in the table 1.

Extending this novel technique of shadow replication, this work extends collocation approach to bring down energy consumption using dynamic cores instead of static cores which shows immense benefits as highlighted in results section.

TABLE 1
COMPARATIVE ANALYSIS OF SHADOW REPLICATION APPROACHES USED IN LITERATURE.

SHADOW REPLICATION APPROACH USED	ENERGY EFFICIENCY ACHIEVED USING		DESCRIPTION	BENEFITS/ENERGY EFFICIENCY ATTAINED	LIMITATIONS
	DVFS	CORES			
Sahdow Computing. [31]	✓		• Introduces shadow replication which balances parameterized tradeoffs between time and hardware redundancy	• Has the ability to save 15%-30% of energy compared to traditional replication.	• Framework not implemented to measure energy savings achieved by this technique.
Energy consumption of resilience mechanisms in large scale systems [32]	✓		• Checks the performance of checkpointing and replication on present and future systems where these approaches are not scalable because of increase in the workload. • Proposes shadow replication with an aim to increase response time compared to checkpointing and traditional replication and at the same time meet power constraints.	• 40% more power savings compared to traditional replication.	• Message synchronization issues between main and shadow processes.
Shadow Computing for High Performance Computing [29]	✓		• Proposes shadow replication as advancement over checkpoint/restart protocols that are time and energy consuming. • Proposes two ways to apply shadow replication namely: energy optimal replication and stretched replication.	• Can save 58% of energy consumed in pure replication. • In the worst case also shadow replication can save 17% of energy consumed.	• Message synchronization issues.
Profit Aware Shadow Replication Approach. [28], [30]	✓		• Proposes profit maximizing, SLA based and energy aware replication called reward based shadow replication. • Aims at maximizing profit for cloud service provider and imposing penalties on violation of SLA. • Also propose profit aware stretched replication for cases when failure detection is not possible.	• Outperforms traditional replication. • Proper tradeoffs achieved between fault tolerance, SLA specification and profit maximization. • Leads to 30% extra profit due to reduced energy consumption.	• Static power hinders energy conservation as when static power is high, slowing down processes does not lead to significant energy savings.
Core level shadow replication Approach. [33], [34]		✓	• Removes the limitations of DVFS approach. • Incorporates the concept of computing cores with shadow replication. • Follows adaptive approach that is it adjusts execution speeds to balance tradeoffs between energy and delay.	• Compared to other energy saving techniques, this approach saves 20% of energy. • Adaptively adapts execution speed to balance tradeoffs between energy and delay.	• Energy savings are limited by static power.

III. AIM OF THE WORK

Given that most of fault tolerance approaches namely check pointing and traditional replication are power consuming in nature. The aim of this paper is to bring forward an approach that not only effectively tolerates faults but is also energy efficient in nature. One such approach is shadow replication.

Mills, Zanti and Melhem were among the pioneers to introduce shadow replication as an energy aware, highly resilient fault tolerance approach. This approach exploited dynamic voltage and frequency scaling [29],[31] to bring about power optimizations. Simulations and experimental evaluations highlighted the advantages of this approach over checkpointing and traditional replication in terms of execution time and energy optimizations. However, DVFS approach associates with it multiple drawbacks that question its viability. Firstly DFVS approach increases the rate of transient faults along with energy optimizations. Secondly, this approach is limited by the availability of number of frequencies and granularity of voltage control.

In order to overcome the disadvantages of DVFS approach, Cui introduced another approach to shadow replication where instead of using DVFS, power is optimized by collocating numerous tasks within one core while keeping the core working at maximum speed. This core level shadow replication undoubtedly enhanced the model proposed by Mills but is not without flaws. Firstly in Cui's model cores are packed within shadow sets. The core level failure in adequately tolerated by this model by making use of replicas placed within shadow core. However, the failure of shadow set is not tolerated and in this kind of failure, the application is entirely rolled back and started from the scratch (Application level failure) which instead of saving energy will require additional energy. Secondly, the tasks are assigned to the cores as and when they come that is in a sequential manner with no regards paid to scheduling that can be energy efficient. Had it been so additional energy would have been saved. Thirdly, this model makes use of only static core allocation policy which restricts the model in terms of energy savings and fault tolerance rate achieved.

This paper appends the shadow replication policy used by Cui to the virtualization environment and proposes appropriate solutions to the problems faced in Cui's approach.

The main contribution of this papers are to design a scheduling approach that assigns tasks to the virtual machines in an energy efficient manner, to tolerate core level and application level failure in an efficient manner, to design an energy aware approach whereby power limits are defined and it is ensured that the virtual machine works under the assigned power ranges, to design energy efficient virtual migration approach and to design dynamic core allocation policy and check for improvement over static core allocation policy in terms of energy consumption, latency, cost and fault tolerance rate attainable.

IV. PROPOSED WORK

This paper integrates shadow replication at virtualization layer. Usually hosts are integrated with virtual machine support to add parallelization to the existing host. Not only this, virtual machines offer easy maintenance, application provisioning, availability and convenient recovery. However, the failures can hamper the successful completion of the application leading to loss of computational data. Usually one way of dealing with this kind of failure is to restart the application from scratch on a new virtual machine. Doing this leads to the loss of work progress and the additional rollbacks increase execution time and henceforth energy consumption. So this is where the need of shadow replication arises. Shadow replication constantly maintained backup of work progress in the shadow core so that when ever application level failure occurs in a virtual machine, the work progress is copied from the shadow core to a new VM without the need to rollback and restart the application. The architecture of the proposed approach is highlighted in the diagram below

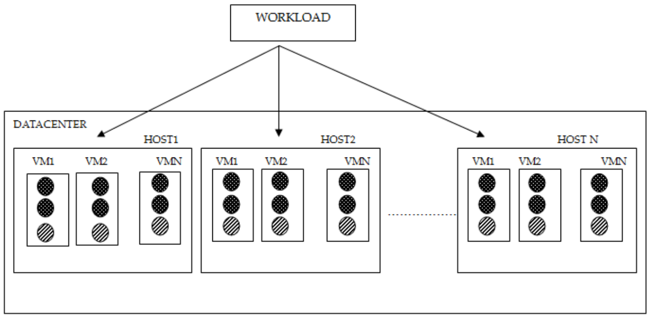

Figure 1: Shadow Replication at Virtual machine level.

● CORE EXECUTING THE MAIN TASK
▨ CORE MAINTAINING REPLICAS/SHADOW CORE

The architecture above describes the datacenter with multiple hosts. Each host is equipped with numerous virtual machines. These virtual machines enable parallelization of work. Further the entire memory capacity of virtual machines is divided among cores of equal size. Of these cores, one core is specifically allocated the job of maintaining replicas that is a shadow core whereas the remaining are the computational cores whereby the workload is executed.

The functionality of the proposed model is explained in detail in the next sub-section.

A. *Scheduling Policy:*

The scheduling policy used in this model is energy efficient in nature. This model makes use of execution rate of virtual machines to schedule the application. Based on the execution speed of VM's, the virtual machines are arranged in such a way that the fastest executing VM will occupy the first place in the list and the VM with least execution capacity will be placed down at the bottom. The datacenter executes varying types of applications that differ in their behavior. Certain applications are compute –intensive while others involve processing large volumes of data. This paper specifically deals with compute intensive applications which require sequential execution of multiple phases with thousands of tasks. Likewise in this paper the compute intensive applications are prioritized based on the number of computation phases. The more computation phases the application exhibits the more priority it holds that is applications are sorted based on their lengths. The greater priority applications are assigned to the VM's with higher execution rates so that greater priority applications are executed faster than others. This scheduling policy will reduce the execution time of an application which in turn will reduce the energy consumption. Furthermore, the application is split up into tasks and assigned to VM's. The number of tasks assigned to each VM depends on the execution rate of each virtual machine. Within the virtual machine, the tasks are equally split among the computational cores. While the computational cores aim at task execution, the shadow core maintains replicas of tasks in execution. The core level replica management saves energy as the use of additional resources is minimized. Moreover core level shadow allocation also overcomes the drawbacks of dynamic voltage and frequency scaling approach.

B. Fault Tolerance:

In this model, failures can occur at core and application level. It is here that shadow replicas can be effectively be used to prevent loss of application progress. Now let us see how failures at each of these levels are successfully tolerated by this model.

1) Core level failures:

In case of core level failures such as core out of memory while executing some task, the replicas of the progress made by the failed task are picked from the shadow core and propagated to the next available core. The next core in addition to executing the tasks at hand will also complete the leftover work of the failed core using the replicas of the failed tasks saved in the shadow core since most of cores remain underutilized.

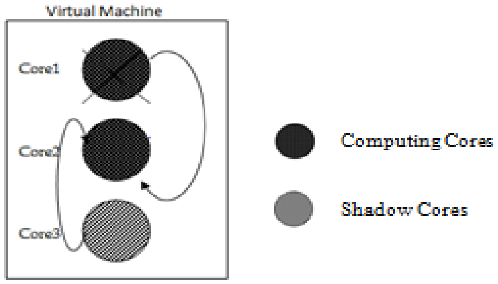

Figure 2: Core Level Failure in Virtual Machine.

In the figure2 above, the virtual machine is equipped with three cores namely core1, core2 and core3. Of these, core1 and core2 are computational cores where the actual task execution takes place and core3 is a shadow core where the replicas of the progress made by the tasks in core1 and core3 are kept. In the case above core1 fails because it eventually gets out of memory while executing a particular task. Now the progress of the failed task in core1 is picked up from shadow core that is core 3 and transferred to core 2 which completes the task in hand along with the failed task of the core1. The important question here is why certain cores run out of memory while other remains underutilized. The simple answer to this riddle is that certain tasks are more memory intensive than others and require greater memory capacity.

2) Application level Failure:

As seen in the case above whenever core level failure occur the replicas of the progress made by the tasks is copied from the shadow core to the next available core but it may so happen that the next core too runs out of memory and so on till the virtual machine eventually runs out of the available computational cores. So this result in the failure of virtual machine to execute the portion of work allotted to it. In the usual case, if this scenario occurs, entire application is rolled back and restarted from the beginning but in the case of the proposed model, the shadow cores are used to copy the progress successfully made before the failure has occurred to a new virtual a machine in the same host. This saves the energy which had to be wasted to roll back the application fully and enhances the performance in cases of such failure. This case is depicted in the figure below:

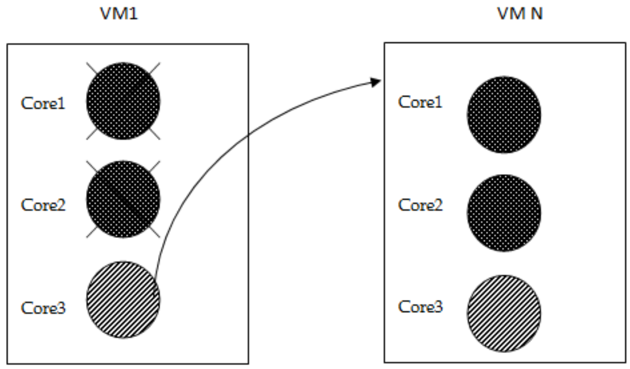

Figure 3: Application Level Failure.

● Computing Cores

▨ Shadow Cores.

The figure above depicts the case of virtual machine level failure whereby both the cores, core1 and core2 in VM 1 runs out of memory and the VM1 is left with no more cores to proceed. So in this case the progress carried before the failure occurred is copied from the shadow core that is core 3 to a new virtual machine.

C. *Maintaining Energy Efficiency:*

The energy consumed by the virtual machines is calculated using the equation 4:

$$\text{Energy Consumption} = \frac{CPU\%}{P1+P2*CPU\%} * (P1+P2) \qquad (4)$$

Here, $CPU\%$ is the percentage of CPU utilization for a given time interval.

$P1$ is the power consumption when the physical host is idle.

P2 is the power consumption when CPU is utilized that is proportional to the overall system workload.

According to the spec benchmark, the energy consumption of the machine cannot exceed 250 watt. So this paper sets 250 watt as a threshold value. If the energy consumption of any virtual machine is found to exceed this value then that machine is considered for migration to a new host. The virtual machine is not migrated within the same host as it may be possible that the other machine too suffers similar problem. Moreover, this energy equation is used to continuously monitor the energy consumption of all the virtual machines. So whenever, the application level failure occurs, the VM is selected based on its energy consumption. The VM with lowest possible energy consumption is selected to proceed with the failed computation.

D. *Dynamic Core Allocation:*

Till now, the entire work on shadow replication was done for static cores which limit energy efficiency and fault tolerance rate achievable. However this paper brings forward the approach of allocating cores dynamically and the benefits it has over static core allocation in terms of energy efficiency, latency, cost and fault tolerance rate achievable. The failures shown in the static approach above are similarly simulated in dynamic core allocation. The results when compared to static approach have shown a significant improvement.

As seen in the application level failure where eventually when all the computing cores run out memory, the computation needs to be shifted to a new virtual machine. This process increases the overhead of moving the workload on a new virtual machine from time to time. In order to minimize this overhead, the cores are allocated dynamically according to the requirements of the workload assigned to the virtual machine. Furthermore in case of core level failure, the VM is equipped with enough cores so that the workload on the failed core gets easily executed by the other cores in the same virtual machine. Apart from these two advantages, dynamic core allocation also prevents the underutilization of cores as only those many cores are allocated that can satisfy the workload requirements. Dynamic core allocation policy is an immense advantage over static core allocation policy in terms of energy efficiency, fault tolerance rate, cost and latency. This is highlighted from the results obtained by simulation presented in the next subsection.

E. *Algorithm:*

- ➢ **I**nput : Cloudlets
- ➢ **O**utput: Latency, Energy Consumed, Fault Rate, Cost.

a. **I**nitialize cloud:
   Including Data centre, hosts, vm's, cores and shadows.

b. **A**ssign Cost with Each vm.

c. Energy Efficient Scheduling:
   - I. **S**ort the vm's according to their execution rates.
   - II. **A**ssign cloudlets to the VM with maximum execution rate which will result in quick computations and lower energy utilization.

d. **C**reate dynamic cores based on cloudlet (workload).

e. **E**xecute cloudlet on vm cores.

f. **C**heck for cores failure due to memory overflow.
   - I. If core fails look for next available computational core.
   - II. Copy the progress from shadow core to the new core.
   - III. The new core completes the unfinished work of the failed core.

g. Check for Application level failure when all the cores in the VM run out of memory.
   - I. In this case look for new VM with lowest energy consumption in the same host.
   - II. Copy the progress from the shadow core to the newly allocated machine.
   - III. Resume the progress on the new VM.

h. **C**heck Energy Consumption according to the equation:
   Energy Consumption =

$$\frac{CPU\%}{P1+P2*CPU\%} * (P1+P2)$$

.

   - I. **I**f Energy > threshold then proactive vm migration is required.
   - II. Energy Efficient VM is selected in a different host.
   - III. **C**opy progress from Shadow core of this VM to the newly selected machine.

i. **O**utput latency, Energy Consumed, Fault Rate, Cost.

F. *Proposed Flowchart*

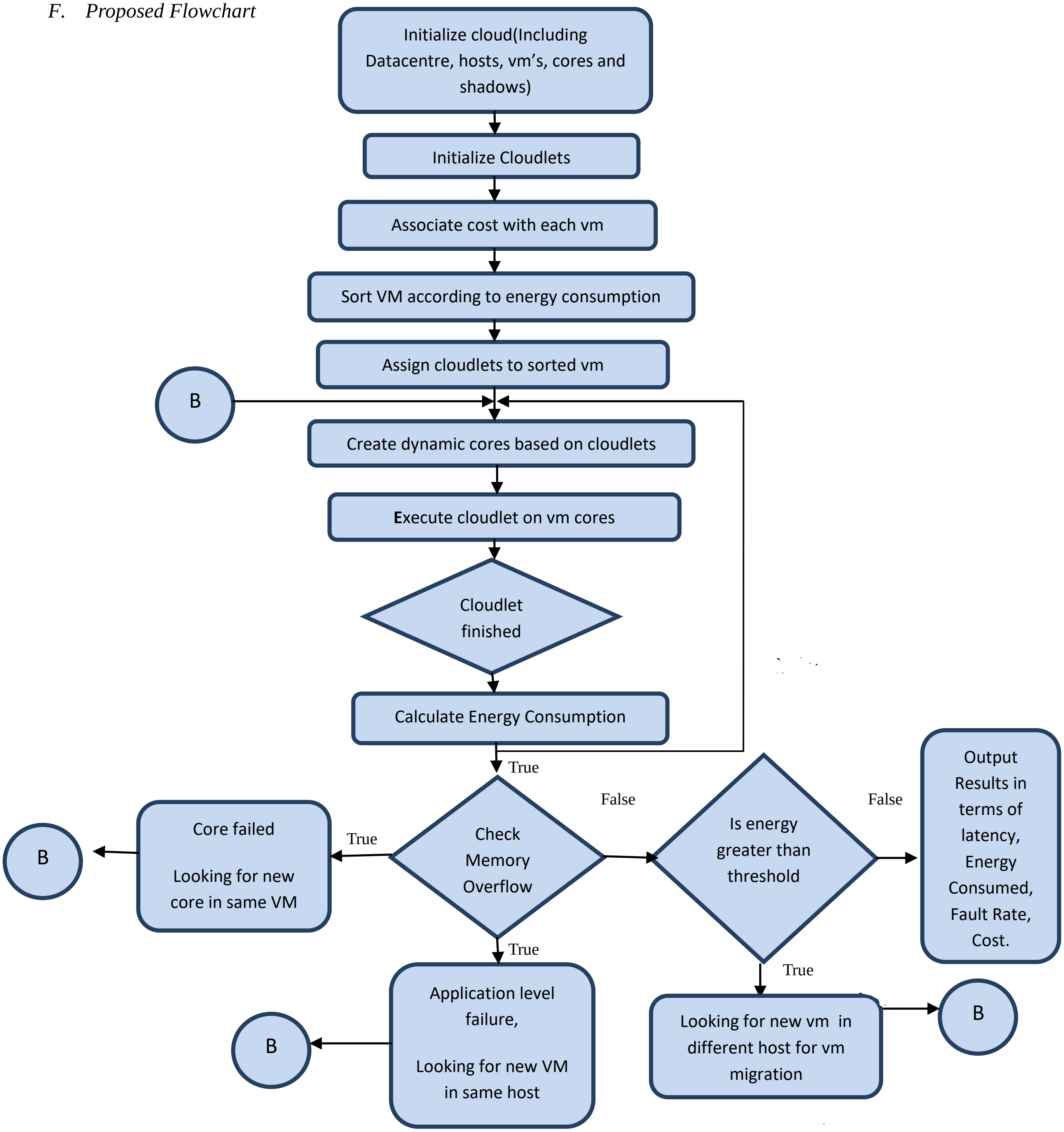

Figure 1: Proposed Flowchart

.

V. RESULTS AND PERFORMANCE ANALYSIS

The results associated with the dynamic core model of the proposed system are compared against static core model and following results have been obtained. The energy consumed is measured in joules, latency in milliseconds, cost in unit cost (rupees, dollars etc) and fault tolerance as the number of faults tolerated.

A. *Cloudlet size 90000*

Parameter	Existing	Proposed
Average energy consumed	40.938152	18.412603
Latency	14096	10755
cost encountered	328	275
fault tolerant rate	69	61

Table 1: Comparison of parameters in case of cloudlet size 9000

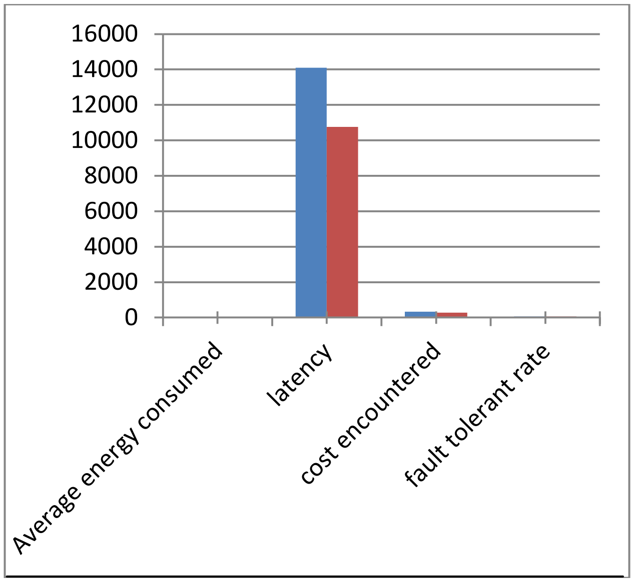

Figure 5: Comparison of parameters with cloudlet size 90000

B. *Cloudlet size 85000*

Parameters	Existing	Proposed
Average energy consumed	54.405746	14.436229
latency	13079	10503
cost encountered	898	159
fault tolerant rate	49	82

Table 2: Cloudlet size 85000 parameter comparison

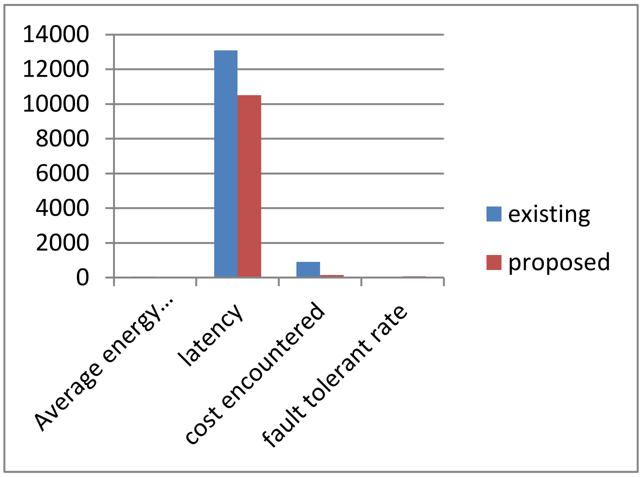

Figure 6: Comparison of parameters with cloudlet size 85000

C. *Cloudlet size 80000*

Parameters	Existing	Proposed
Average energy consumed	51.703804	14.504681
latency	11532	9038
cost encountered	793	276
fault tolerant rate	49	83

Table 3: Cloudlet size 80000 parameter comparison

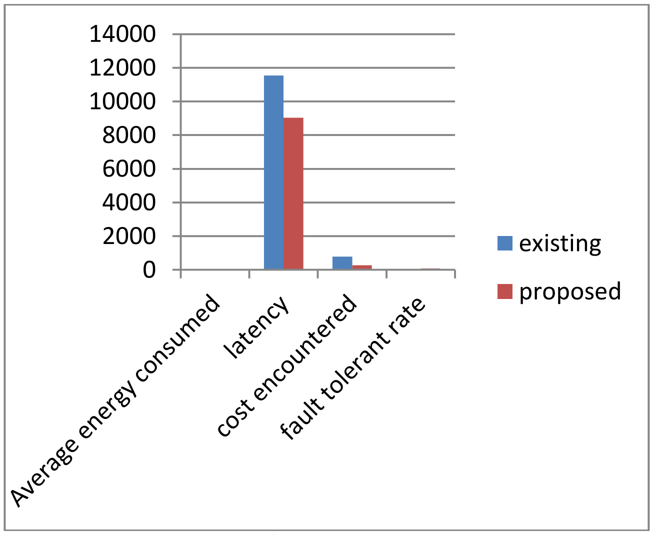

Figure 7: Comparison of parameters with cloudlet size 80000

D. *Cloudlet size 75000*

Parameters	Existing	Proposed
Average energy consumed	46.09626	25.330738
latency	14574	12533
cost encountered	637	164
fault tolerant rate	54	79

Table 4: Cloudlet size 75000 parameter comparison

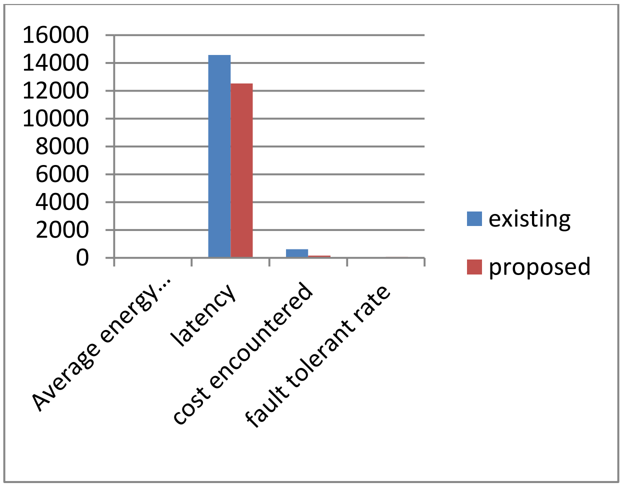

Figure 8: Comparison of parameters with cloudlet size 75000

E. *Cloudlet size 70000*

Parameters	Existing	Proposed
Average energy consumed	54.82179	5.2413588
latency	11893	18173
cost encountered	751	157
fault tolerant rate	51	76

Table 5: Cloudlet size 70000 parameter comparison

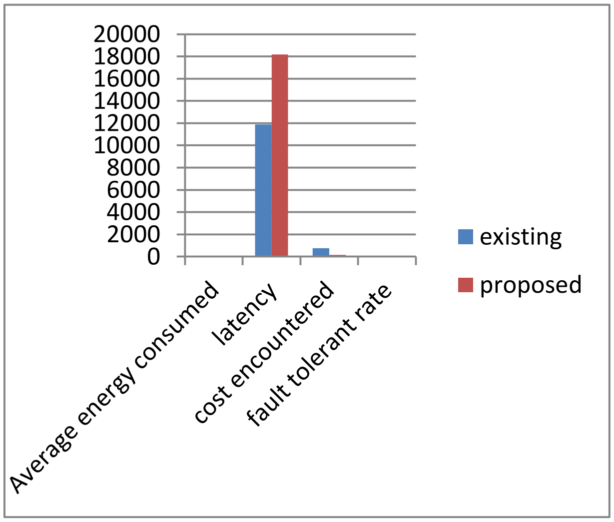

Figure 9: Comparison of parameters with cloudlet size 70000

Results of the proposed system that is shadow replication using dynamic core allocation in virtual machines shows significant improvement in terms of energy efficiency, fault tolerance rate, cost and latency when compared to static core allocation policy proving worth of the study.

VI. CONCLUSION

The proposed model **"Shadow Replication using Dynamic Core Allocation for Application Fault Tolerance in Virtual Environment"** is energy efficient, cost saving, latency conserving and highly resilient approach. The result so obtained proves the worth of the proposed model when compared to the static core allocation policy.

The proposed model avoids the repeated shifting of workload to a new virtual machine in cases of failure by allocating cores dynamically based on the workload pattern which provides extra slack to tolerate failures making it high fault tolerant and latency saving approach. Moreover as cores are allocated dynamically, only those many cores are allocated that can satisfy the workload requirements preventing underutilization of cores and hence saving cost. Energy efficient allocation and reallocation, keeping replicas in the shadow core making this approach highly energy efficient.

Seeing the advantages of proposed approach, the future work is to test the model for different kinds of faults and to study dynamic core allocation in depth according to varying patterns of workload

REFERENCES

[1] Meroufel and Belalem, G., (2014). "Adaptive time-based coordinated checkpointing for cloud computing workflows," *Scalable Comput.*, vol. 15, no. 2, pp. 153–168.

[2] Kliazovich, D., Bouvry, P., and Khan, S. U., (2012). "GreenCloud : A Packet-level Simulator of Energy- aware Cloud Computing Data Centers," *J. Supercomput.*, vol. 62, no. 3, pp. 1263–1283,

[3] Milani, B. A. and Navimipour, N. J., (2016). "A comprehensive review of the data replication techniques in the cloud environments: Major trends and future directions," *J. Netw. Comput. Appl.*, vol. 64, pp. 229–238.

[4] Zhu, Q. and Zhou, Y., (2005). "Power-Aware Storage Cache Management." *IEEE Trans. Computers*, vol. 54, no. 5, pp. 587-602 .

[5] Brown, R., Masanet, E., Nordman, B., Tschudi, B. , Shehabi, A., Stanley, J., Koomey, J., Sartor, D., Chan., P., (2008). Report to Congress on Server and Data Center Energy Efficiency (LBNL-363E), public law 109–431, Lawrence Berkeley National Laboratory, Berkeley, CA .

[6] Aupy, G., Benoit, A., Diouri, M. E. M., Glück, O., Lefèvre, L.,(2015)."Energy-Aware Checkpointing Strategies," In *Fault-Tolerance Techniques for High-Performance Computing*, pp. 279-317, Springer International Publishing, pp. 279–317.

[7] Diouri, M. E. M., Gluck, O. , Lefevre, L. and Cappello, F., (2013). "ECOFIT: A Framework to Estimate Energy Consumption of Fault Tolerance Protocols for HPC Applications," *Proc. IEEE/ACM thirteenth Int. Symp. Cluster, Cloud and Grid Computing (CCGrid 2013)*, pp. 522–529.

[8] Diouri, M. E. M., Glück, O., Lefèvre, L. and Cappello, F.,(2013). "Towards An Energy Estimator for Fault Tolerance Protocols," J. *ACM SIGPLAN Notices*, vol. 48, no. 8, pp. 313-314.

[9] Ibtesham, D., Debonis, D., Arnold, D. and Ferreira, K. B., (2014). "Coarse-Grained Energy Modeling of Rollback/Recovery Mechanisms," *Proc. IEEE/IFIP Forty-forth Annu. Int. Conf. Dependable Systems and Networks (DSN 2014)*, pp. 708–713.

[10] Diouri, M. E. M., Gluck, O., Lefevre, L. and Cappello, F., (2012). "Energy Considerations in Checkpointing and Fault Tolerance Protocols," *Proc. IEEE forty-two- Int. Conf. Dependable Systems and Networks Work. (DSN-W 2012)*, pp. 3–8.

[11] Mills, B. N., Grant, R. E., Ferreira, K. B. and Riesen, R., (2013).

“Evaluating Energy Savings for Checkpoint/Restart,” *Proc. first Int. Work. Energy Efficient Supercomputing (E2SC 2013),* pp. 61–68.

[12] Saito, T., Sato, K., Sato, H., and Matsuoka, S., (2013). “Energy-Aware I/O Optimization for Checkpoint and Restart on a NAND Flash Memory System,” *Proc. Third Work. Fault-tolerance for HPC at Extreme scale (FTXS ’13),* pp. 41–48.

[13] Helary, J. M., Mostefaoui, A., Netzer, R. H. B. and Raynal, M. , (1997). “Preventing Useless Checkpoints in Distributed Computations,” *Proc. IEEE sixteenth Symp. Reliable Distributed Systems*, pp. 183–190.

[14] Sayed, N.E. and Schroeder, B., (2014). “To Checkpoint or Not to Checkpoint: Understanding Energy-Performance-I/O Tradeoffs in HPC Checkpointing,” *Proc. IEEE Int. Conf. Cluster Computing (CLUSTER 2014),* pp. 93–102, 2014.

[15] Aupy, G., Benoit, A., Melhem, R., Renaud-Goud, P. and Robert, Y., (2013). “Energy-Aware Checkpointing of Divisible Tasks with Soft or Hard Deadlines,” Proc. IEEE *Int. Conf. Green Comput. (IGCC 2013), pp.1-8*.

[16] Ananthanarayanan, G., Agarwal, S., Kandula, S., Greenberg, A., Stoica, I., Harlan, D., Harris, E., (2011). "Scarlett: coping with skewed content popularity in mapreduce clusters." *Proc. ACM sixth Conf. Computer systems*, pp. 287-300..

[17] Abad, C. L., Lu, Y. and Campbell, R. H., (2011). “DARE: Adaptive data replication for efficient cluster scheduling,” *Proc. IEEE Int. Conf. Cluster Computing (ICCC)*, pp. 159–168.

[18] Ranganathan, K., Iamnitchi, A. and Foster, I., (2002). “Improving data availability through dynamic model-driven replication in large peer-to-peer communities,” Proc. *IEEE/ACM second Int. Symp. Cluster Computing and the Grid (CCGrid 2002)*, pp. 0–5.

[19] Chang, R.S., Chang, H.P., Wang, Y.T., (2008). "A dynamic weighted data replication strategy in data grids." Proc. *IEEE/ACS Int. conf. Computer Systems and Applications (AICCSA 2008)*, pp. 414-421.

[20] Weddle, C., Oldham, M. , Qian, J. I. N., Wang, A. A., Reiher, P. and Kuenning, G., (2007). “PARAID : A Gear-Shifting Power-Aware RAID,” *ACM Trans. Storage (TOS)* , vol. 3, no. 3.

[21] Kim, J. and Rotem, D.,(2012). “FREP: Energy proportionality for disk storage using replication,” *J. Parallel and Distributed Computing,* vol. 72, no. 8, pp. 960–974.

[22] Wan, J., Yin, C., Wang, J., Xie, C., (2012). “A new high-performance, energy-efficient replication storage system with reliability guarantee,” Proc. *IEEE Symp. Mass Storage Systems and Technologies (MSST), pp.1-6.*

[23] Boru, D., Kliazovich, D., Granelli, F., Bouvry, P. and Zomaya, A. Y., (2015). “Energy-efficient data replication in cloud computing datacenters,” J. *Cluster Compuing* ., vol. 18, no. 1, pp. 385–402.

[24] Boru, D., Kliazovich, D., Granelli, F., Bouvry, P. and Zomaya, A. Y. (2013). “Energy-efficient data replication in cloud computing datacenters,” Proc. IEEE Work. *Globecom Work. (GC Wkshps),* pp. 446–451.

[25] Lin, Y. and Shen, H., (2017). “EAFR: An Energy-Efficient Adaptive File Replication System in Data-Intensive Clusters,” *IEEE Trans. Parallel and Distributed Systems*, vol. 28, no. 4, pp. 1017–1030.

[26] Kaushik, R. T., (2010). “GreenHDFS : Towards An Energy-Conserving , Storage-Efficient , Hybrid Hadoop Compute Cluster,” Proc. USENIX Int. conf. Power aware computing and systems (HotPower'10), pp. 1–9.

[27] Lang, W., Patel, J. M. and Naughton, J. F., (2010). “On energy management, load balancing and replication,” J. *ACM SIGMOD Record*, vol. 38, no. 4, pp.35-42.

[28] Cui, X., Mills, B. , Znati, T. and Melhem, R., (2014). “Shadow replication: An energy-aware, fault-tolerant computational model for green cloud computing,” J. *Energies*, vol. 7, no. 8, pp. 5151–5176.

[29] Mills, B., Znati, T., Melhem, R. and Ferreira, K. B., (2013). “Shadow Computing An Energy-Aware Resiliency Scheme for High Performance Computing,” Sandia National Laboratories (SNL-NM), Albuquerque, New Mexico, United States.

[30] Cui, X., Mills, B., Znati, T. and Melhem, R., (2014). “Shadows on the cloud: An energy-aware, profit maximizing resilience framework for cloud computing,” Proc. Forth In*t. Conf. Cloud Computing and Services Science,* pp. 15–26.

[31] Mills, B., Znati, T., and Melhem, R., (2014). "Shadow computing: An energy-aware fault tolerant computing model." Proc. IEEE Int. Conf. *Computing, Networking and Communications (ICNC),* pp. 73-77.

[32] Mills, B., Znati, T., Melhem, R., Ferreira, K. B. and. Grant, R. E , (2014). “Energy consumption of resilience mechanisms in large scale systems,” *Proc. Twenty-two Euromicro Int. Conf. Parallel, Distributedand. Network-Based Processing (PDP 2014)*, pp. 528–535, 2014.

[33] Cui, X., Znati, T., and Melhem, R., (2016). “Adaptive and Power-Aware Resilience for Extreme-scale Computing.” Proc. IEEE Conf. *Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)*, pp. 671-679.

[34] Cui, X., (2016) “ADAPTIVE AND POWER-AWARE FAULT TOLERANCE FOR FUTURE EXTREME-SCALE COMPUTING” P.hd dissertation , Computer Science Department, University of Pittsburgh , Pennsylvania, America.