International journal Of Business Management
ISSN NO. 2349-3402
Available at www.ijbm.co.in
VOL. 3 Issue 1, 2017

A Novel Technique for Handling Overloading in Distributed System

Arshpreet Kaur[1]

Abstract:

Distributed systems are built up on top of existing networking and operating systems software. A distributed system comprises a collection of autonomous computers, linked through a computer network. The whole task is divided on number of resources. The user is not aware that the jobs are executed by multiple computers. Fault tolerance is the ability of a system to withstand failure and continue to provide services in the event of an error. Fault tolerance systems are designed to ensure that in the event of a failure, crash, or a major user error, data is not lost and the system can continue to provide its services, thereby increasing the reliability and dependability of a system usually by masking the software and hardware faults. Overloading problem occurs due to load imbalance in Distributed Computing System. This problem leads to the fault occurrence problem and degrades network performance. The existing algorithm is the master and slave architecture through which the task are assigned by the master node to slave nodes and slave nodes execute task as assigned by the master nodes. In such type of architecture, the divide and rule technique is followed, and after executing the task slave nodes will revert back to master nodes. The major problem in this architecture is of fault, if one slave node get failed the task allocated by master node will not get completed and fault occurred. To remove the problem of overloading genetic algorithm has been applied upon the candidate nodes.

Keywords: distributed computing system, fault tolerance, task reallocation and load balancing.

1. Assistant Professor in Chandigarh Group of Colleges Jhanjeri, Mohali.

1) Introduction

A distributed system is a collection of independent computers that appear to the users of the system as a single coherent system. Distributed system is an application that executes a collection of protocols to coordinate the actions of multiple process on a communication networks so that all the available components are combined together to perform a single or small set of related tasks. Through the communication networks, distributed computer can be able to access remote resources as well as local resources.

The existence of the multiples users are transparent in distributed computing systems. Moreover it is defined that no single computer takes load of resources [9]. The whole task is divided on number of resources. The user is not aware that the jobs are executed by multiple computers subsist in remote locations. Distributed systems are built up on top of existing networking and operating systems software. A distributed system comprises a collection of autonomous

International journal Of Business Management
ISSN NO. 2349-3402
Available at www.ijbm.co.in
VOL. 3 Issue 1, 2017

computers, linked through a computer network and distribution middleware.

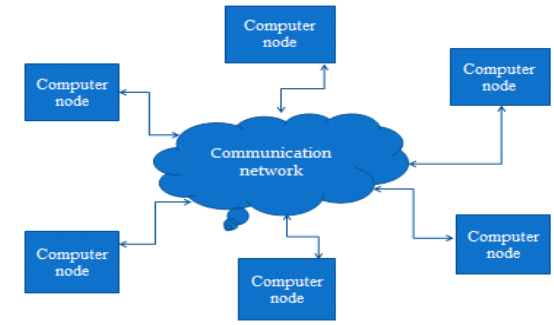

Fig. 1: Distributed Computing System

Distributed Computing system is a heterogeneous in nature. There are various hardware and software required to build a distributed system. Every node has different characteristics which are participated in a distributed computing system. Distributed Computing is an infrastructure that provides access globally to the information and heterogeneous computing resources.
In today's computing world people does not want to wait for his order or request completion for long period. They desire to get their result or solution at instant or as soon as possible anytime and anywhere. A mobile computing is an activity performed on a spatially distributed system. These networked computers may be in the same room, same campus, same country, or in different continents [10]. In distributed computing, individual users can access computers without having to consider location, operating system, account administration and other details.

2) Distributed Computing Architecture

In distributed computing infrastructure resources are come from and associated with physically scattered administrative domains to make available the various resources to the users collectively. In a system, computing nodes can be independently operated from different locations because main node is not physical place at one place. Each computer on the grid is a distinct computer [6]. To work out a common problem several grids are clustered together to work upon them. There are five layers in grid computing. These are as follow:
a. Fabric Layer: Fabric layer is present at the bottom of the layered architecture. It provides shareable resources such as CPU time, network bandwidth, scientific instruments, memories, like sensors, telescope, etc. Data received by sensors at this layer can be stored in the database over grid and can be transmitted directly to other computational nodes. There are standard grid protocols are responsible for resource control accomplishment of sophisticated sharing operation is the measure for quality of this layer. Operating system, queuing systems and processing kernels also form the part of this layer.

International journal Of Business Management
ISSN NO. 2349-3402
Available at www.ijbm.co.in
VOL. 3 Issue 1, 2017

b. Connectivity Layer: Secure and easy access protocols are placed in this layer. Protocols which are associated with communication and authentication required for transactions are placed in this layer. These communication protocols authorize the exchange of data between resource layer and fabric layer. Authentication protocols are meant to provide secure cryptographic mechanisms for identification of users and resources.

c. Resource Layer: This layer specifies operating protocols with shared resources. Resource layer build on the connectivity layer's communication and authentication protocols to define Application Program Interfaces (API) and software development kit (SDK) for secure negotiation, accounting, initiation, control, monitoring and payment of sharing resources.

d. Collective layer: This layer consists of general purpose utilities. In the shared resources collaborative operations are placed in this layer and it coordinates sharing of resources like, collocation, directory services, brokering services, and scheduling, monitoring and diagnostic services.

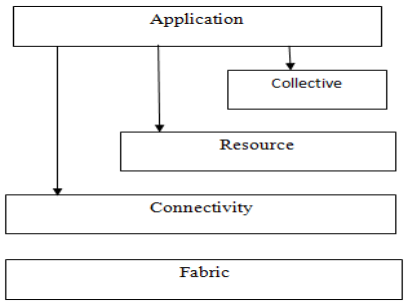

Fig. 2 : Architecture of Distributed Computing

e. Application layer: It is the top layer of the grid architecture. This layer consists of application which the user will put into practice. Moreover, this layer provides interface to the users and administrators to interact with the grid.

3) Fault Tolerance

In Distributed system we have multiple software and hardware components that provide inexpensive methods to accesses high end computations. Any mistake in distributed system can cause a system into collapse if not properly detected and recovered at time. Fault tolerance can be a computer system or a backup component or a procedure that immediately take place of the fault component, with no lass of service. Fault-tolerance is the important methods that provide continue reliability by applying extra hardware like processors, resource, and communication links. When distributed system has faults in software system than to deal with that faults massages are to be added to the system. The faults can be classified on several factors such as:
Fault Tolerance can be achieved with the help of two ways. These ways are as follow:

International journal Of Business Management
ISSN NO. 2349-3402
Available at www.ijbm.co.in
VOL. 3 Issue 1, 2017

- **Recovery:** If distributed system has any faults then it should be recovered easily without any loss. Faults like cutting wire, power breakdown, hardware fault, and fault in CPU, storage fault can be easily recovered but error in software like response failure, timing failure, Operating system crashes are difficult to recover.
- **Redundancy:** Distributed system provides the equivalent services, so if one service does not work, then the execution of task does not failed because we have multiple parallel machines and this redundancy does not be very expensive.

There are several types of faults or abnormal situations that can affect the services of grid environment. These include:

a. Network Faults: Faults those results due to network partition, packet corruption.

b. Physical Faults: Faults that occur in CPU, in memory, and in storage devices.

c. Life-Cycle Faults: Versioning faults.

d. Processor Faults: Machine or operating system crashes.
Process Faults: Resource shortage, software bug.

e. Service Expiry Fault: The service time of a resource may expire while application is using it.

f. Interaction Faults: Protocol incompatibilities, Security incompatibilities, Policy problems, Timing overhead.

4) Overloading in Distributed System

Distributed computing is a network of many independent computers, each performs a portion of an overall task, to achieve a computational result much more quickly than with a single computer. In distributed system, the master node divides the overall task into multiple sub- nods and these sub-nodes complete the task according to the given instruction of the master node but sometimes, when sub-nodes have insufficient number of resources then they becomes overloaded and fault occurred, then that tasks are reallocated to the other sub-nodes means Imbalance in the network creates overloading problem in distributed computing system. Further overloading is responsible for the network performance degradation. Therefore a good study is requires to develop a good fault tolerant system [14]. By applying extra hardware like processors, resource, communication links hardware fault tolerance can be achieved. In software fault tolerance tasks, to deal with faults messages are added into the system [16].

International journal Of Business Management
ISSN NO. 2349-3402
Available at www.ijbm.co.in
VOL. 3 Issue 1, 2017

5) Problem Statement

The main importance of distributed computing system is that in which master node divide, the task into multiple tasks, the slave's nodes execute task on the behalf of master node. When the task scheduling and execution of tasks performs over large scale then distributed systems plays an important role on achieving good performance and high system utilization. In Distributed system we have multiple software and hardware components that provide inexpensive methods to accesses high end computations. There are various issues in distributed computer system, with the dramatic increase in users of distributed system; the complexity of the network considerably increases so there is a need to develop efficient job schedulers. The goal of a job scheduling system is to efficiently manage the distributed computing power of workstations, servers, and supercomputers in order to maximize job throughput and system utilization.
When the task is equally divided among the nodes then it enhances the network efficiency and reduces the task execution time. On the basis of capacities of processors and communication links, we allocate the tasks among processors. When the task is not equally divided among the nodes, chance of error occurrences will be increased and the node may be failed. A node failure problem may be occurs due to overloading of the node. To handle the fault which occurs due to overloading of node, reallocation of the task,to the most reliable candidate node needs to be done.

6) Literature Survey

M. Chetepen et.al [5] provided some scheduling heuristics based on job replication and rescheduling of failed jobs. Their heuristics do not depend on particular grid architecture and they are suitable for scheduling any application with independent jobs. Scheduling decisions are based on dynamic information on the grid status and not on the information about the scheduled jobs.

T. Altameem et.al [10] described about fault tolerance in distributed computing that considered fault tolerance is a crucial issue in grid computing. Fault tolerance can enhance grid throughput, utilization, response time and more economic profits. All mechanisms proposed to deal with fault-tolerant issues in grids are classified into: job replication and job check pointing techniques. These techniques are used according to the requirements of the computational grid and the type of environment, resources and virtual organizations it is supposed to work with. Each has its own advantages and disadvantages which forms the subject matter.

F. G. Khan et.al [3] presented a performance evaluation of most commonly used faulttolerant techniques (FTTs) in grid computing. These FTTs include retrying, checkpointing, alternate resource and alternate task. The metrics used in their evaluation are processing time, throughput and turnaround time, waiting time and network delay, failure rate, execution time.

International journal Of Business Management
ISSN NO. 2349-3402
Available at www.ijbm.co.in
VOL. 3 Issue 1, 2017

Qin Zheng et.al [7] presented an efficient dynamic scheduling and discussed dynamic planning and adaption of task. Their objective is to optimize the response time while limiting the number of runtime adaption and considered both cases with and without offline planned schedules. Adaption has made on the basis of current information about the start &completion time of its tasks.

B. Naziret et.al [2] presented an adaptive fault tolerant job scheduling strategy for grids. Their strategy is checkpointing-based. It maintains the fault index of grid resources. The scheduler makes scheduling decisions according to the value of the fault index of resources and response time of resources.

Ajay Jangra et.al [1] proposed an Adaptive Ranking Task Scheduling (ARTS) algorithm for scheduling the tasks in grid environment. Objective of proposed work is to minimize the average completion time of submitted tasks as compared with other scheduling methods. Internally ARTS draw up two important basic factors concerned with type of grid lets whether data centric or computation centric and assigning tasks to the resources on the basis of organization rank (OR) value in a grid environment.

JasmaBalasangaMeshwaraet.al [4] presents a load-balancing algorithm by the strong points of neighbour-based and cluster-based load-balancing methods. Then they integrate the proposed load-balancing approach with fault-tolerant scheduling namely MinRC and develop a performance-driven fault-tolerant load-balancing algorithm or PD_MinRC for independent jobs. In order to improve system flexibility, reliability, and save system resource, PD_MinRC employs passive replication scheme. Their main objective is to arrive at job assignments that could achieve minimum response time, maximum resource utilization, and a well-balanced load across all the resources involved in a grid or a cluster.

Yagoubi.B et.al [11] proposed a model to demonstrate grid architecture and an algorithm to schedule tasks within grid resources. The algorithm tries to distribute the workload of the grid environment among the grid resources, fairly. The strategies used to create load balancing with in grid resources improve the throughput of the whole grid environment and proper utilization of grid resources is achieved.

U. Karthick Kumar [14] addresses the fairness issues by using mean waiting time. It scheduled the task by using fair completion time and rescheduled by using mean waiting time of each task to obtain load balance. This algorithm scheme tries to provide optimal solution so that it reduces the execution time and expected price for the execution of all the jobs in the grid system is minimized. The performance of the proposed algorithm compared with other algorithm by using simulation

P K Yadavet.al[17] explained that in Distributed computing systems (DCSs), task allocation

International journal Of Business Management
ISSN NO. 2349-3402
Available at www.ijbm.co.in
VOL. 3 Issue 1, 2017

strategy is an essential phase to minimize the system cost. To utilize the capabilities of distributed computing system (DCS) for an effective parallelism, the tasks of a parallel program must be properly allocated to optimal solution to the given problem.

7) Proposed Methodology

Distributed computing system consist of various autonomous computers, each computer perform some part of whole task, to get the result more quickly as compared to single computer. When the task scheduling and execution of tasks performs over large scale then distributed systems plays an important role on achieving good performance and high system utilization, but overloading is the one of the problem in the Distributed system. In distributed system the master node divide, overall task into slave nodes but when the candidate node overloaded during task execution then the reallocation of the task needs to be done.

System Framework

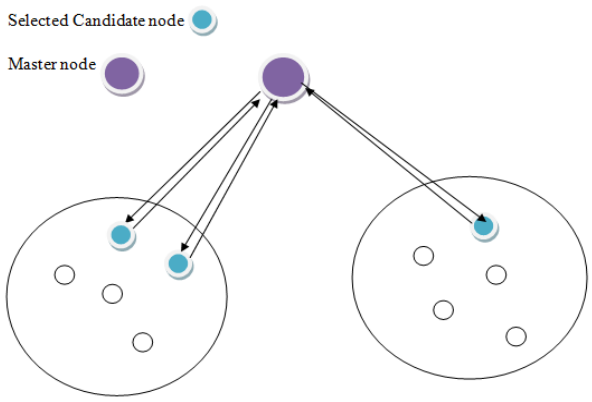

Fig. 3: Master node with Candidate nodes

There is a one master node in the network. After that there are many candidates in the network which operate all other nodes and form clusters. Master node pings candidate nodes. The nodes which are free from overloading reply back to the master node. In above figure first node fails to reply back because of overloading.

International journal Of Business Management
ISSN NO. 2349-3402
Available at www.ijbm.co.in
VOL. 3 Issue 1, 2017

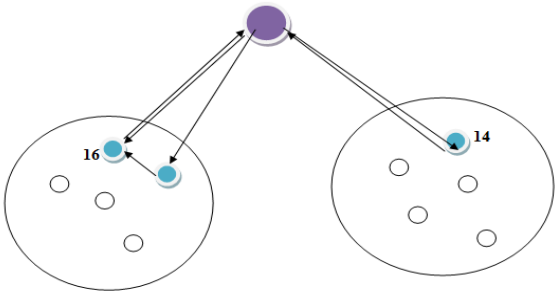

Fig. 4 : Choose Candidate Node according to the Probability

To remove the problem of overloading, genetic algorithm has been applied upon the candidate nodes. In this technique, when the nodes are overloaded then the probabilities of the selected candidate nodes are calculated.

The candidate node which has the highest probability will be selected for the reallocation of the uncompleted task. After the reallocation the selected node will execute the task and the results are returned back to the master node.

8) Purposed algorithm

1. Start
2. Define Distributed system with defined number of nodes.
3. Enter number of task to be executed.
4. Enter the master node time.
5. Enter maximum execution time and failure rate of the master node
6. Enter the failure rate and execution time of each candidate node.
7. Select best candidate node for the allocation of the task.
8. To select best candidate node comparison is made between master node and each candidate node.
9. Candidate nodes which have less failure rate and execution time than master node will be selected as best nodes for task execution.
10. Selected candidate nodes starts, execution of the task.
11. If candidate node is overloaded during the execution of task then fault is occurred and the task is not completed.
12. Calculate probability of selected candidate node.
13. Reassign the task to candidate node with highest probability.
14. Recovery node starts execution of the task.

International journal Of Business Management
ISSN NO. 2349-3402
Available at www.ijbm.co.in
VOL. 3 Issue 1, 2017

9) FLOW CHART

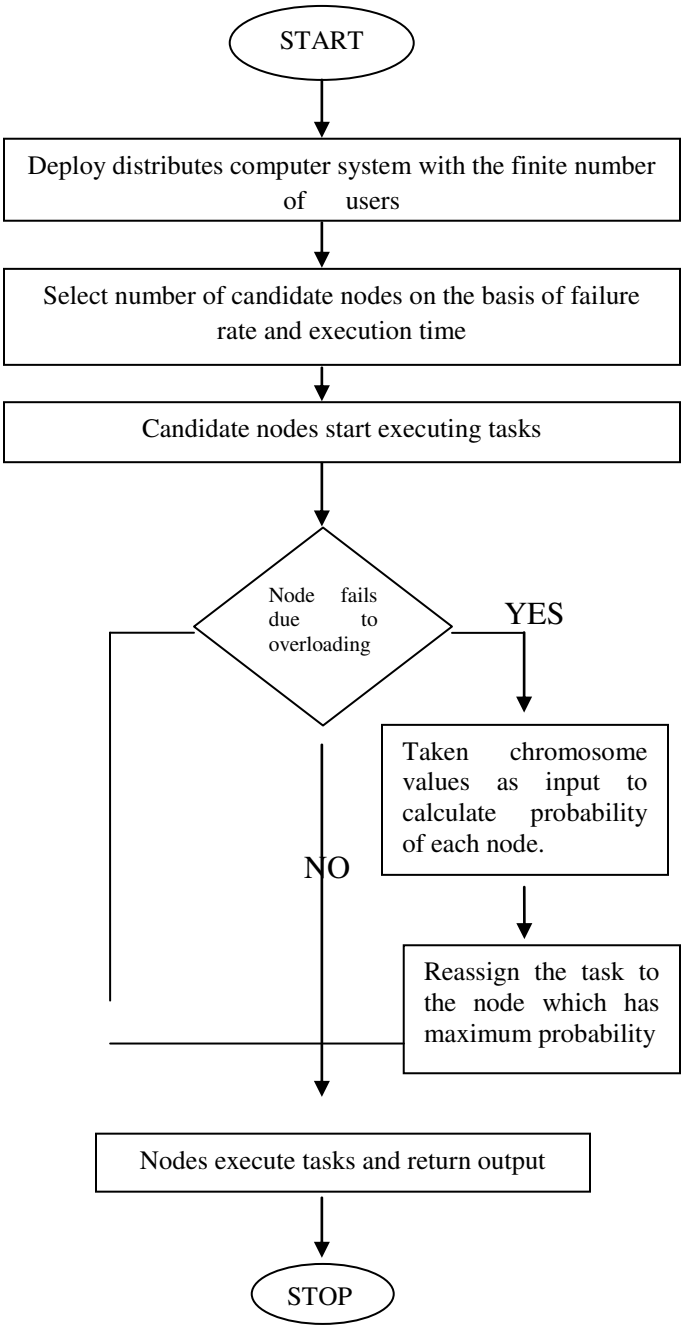

Fig. 5: Flowchart

International journal Of Business Management
ISSN NO. 2349-3402
Available at www.ijbm.co.in
VOL. 3 Issue 1, 2017

10) Experimental results

The results of existing technique that is Monte-Carlo and after enhancing the Monte-Carlo by using the genetic algorithm have been shown. The comparison has been made between existing and proposed techniques on the basis of parameters i.e. Processing time, resource consumption, energy consumption.

a. Processing time

Table 1 : Comparison table of Processing time

Number of tasks	Processing time (seconds)	
	Existing algorithm	Proposed algorithm
8	30	22
10	45	37
12	52	42

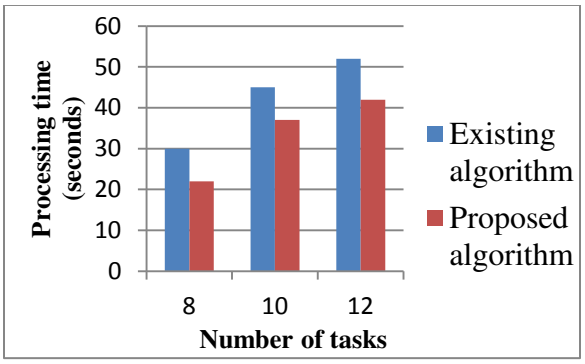

Fig. 6 : Processing Time graph

in figure 6 , the processing time of existing scenario in which fault is occurred due to node overloading is shown and processing time of proposed algorithm is shown in which fault is recovered through genetic algorithm. In the existing scenario fault is occurred and master node is keep on waiting for the task completion. The genetic algorithm will reassign the task to most appropriate node and it will complete the task due to which processing time of proposed algorithm is less as compared to existing scenario. Each time when user enters different number of tasks, processing time of proposed as well as existing algorithm gets changed, but for each input, the processing time of the proposed algorithm will be better than the existing algorithm.

International journal Of Business Management
ISSN NO. 2349-3402
Available at www.ijbm.co.in
VOL. 3 Issue 1, 2017

b. Resources consumption

Table 2: Comparison table of Resource Consumption

Number of tasks	Resource consumption (bytes)	
	Existing algorithm	Proposed algorithm
8	15	11
10	18	13
12	22	15

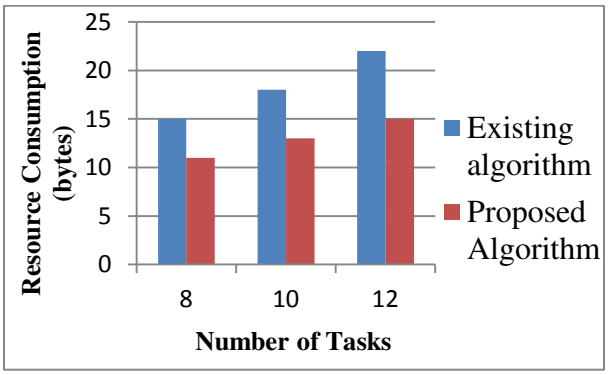

Fig. 7: Resources Consumption graph

As shown in figure 5.12, the resource comparison is shown in terms of buffer size. In the existing and present scenario resources are assigned to candidate nodes for task execution. Due to the fault occurrence in the existing scenario the task will not be completed so the resource consumption is more as compared to the proposed algorithm where the fault is removed by reallocating the task. Each time when user enters different number of tasks, resource consumption of proposed as well as existing algorithm gets changed.

c. Energy consumption

Table 3: Comparison table of Energy Consumption

Number of tasks	Energy consumption (joules)	
	Existing algorithm	Proposed algorithm

International journal Of Business Management
ISSN NO. 2349-3402
Available at www.ijbm.co.in
VOL. 3 Issue 1, 2017

8	120	100
10	145	130
12	160	142

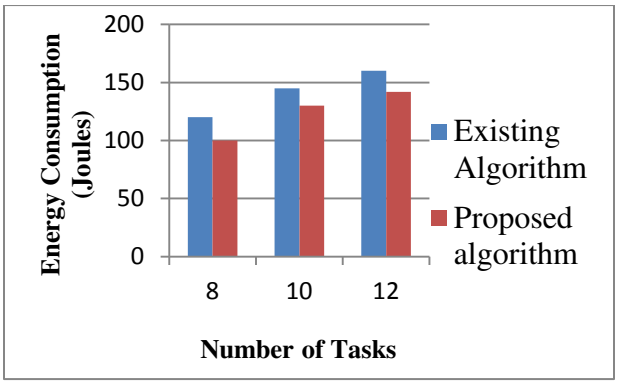

Fig. 8: Energy Consumption graph

As shown in figure 8, the energy comparison is made between the existing and proposed scenario. It is been analyzed that when fault is occurred in the network, master node is keep on waiting for the task to complete due to which energy consumption of the network is high. When task is re-assigned to other candidate node, tasks get completed on time due to which, energy consumption of the network is less. Due to the increases in processing time the energy consumption also increases and vice versa.

11) Conclusion

Distributed computer system is a consolidation of multiple and self-governing computers. In this, user sends the assignment to the master node and then overall assignment is divided into smaller and independent computers, to achieve the final result more rapidly than an individual computer but when sub-nodes have insufficient number of resources then they becomes overloaded and fault occurred, then that tasks are reallocated to the other sub-nodes. Imbalance in the network creates overloading problem. Further overloading is responsible for the network performance degradation.

The existing technique has a limitation of fault, if one slave node gets failed or overloaded, the task allocated by master node will not get completed and fault is occurred. In this thesis, a novel technique has been proposed which overcome the fault of overloading.

To solve the Overloading problem, genetic algorithm has been applied to the candidate node and entire task will be reassigned to the new selected candidate node. This approach will enhance the network performance, reduce execution time and less battery consumption. The proposed

International journal Of Business Management
ISSN NO. 2349-3402
Available at www.ijbm.co.in
VOL. 3 Issue 1, 2017

algorithm is based on the number of task, failure rate, execution time and the time taken by the master node for fault recovery.

12) Future Work

In future, enhancement in the proposed algorithm will be made to handle certain security attacks in overloaded distributed systems; these attacks are denial-of-service attacks which reduces the networks reliability and efficiency.

References

[1] A. Jangra, A.Kumar, "Dynamic Prioritization Based Efficient Task Scheduling for Grid Computing", IEEE International Conference on Information Management in the Knowledge Economy, vol. 4, pp. 978-981, December2013.

[2]B. Nazir, K. Qureshi, F. G. Khan, "AdaptiveCheckpointing Strategy to Tolerate Faults in Economy Based Grid", Journal of Supercomputing, vol. 50, no. 1,pp. 1-18, October 2009.

[3] F. G. Khan, K. Qureshi, B. Nazir, "Performance Evolution of Fault Tolerance techniques in Grid Computing System",Journal of Computing and Electrical Engineering, vol. 36, no. 6, pp. 1110-1122, November2010.

[4] J. B. Meshwara, N. Raju, "Performance-Driven Load Balancing with a Primary-Backup Approach for Computational Grids with Low Communication Cost and Replication Cost", IEEE Transactions on Computers, vol. 62,no. 5, pp. 990-1003, May 2013.

[5] M. Chtepen, B. Dhoedt, F. Cleays, P. Vanrolleghem, "Evaluation of Replication and Rescheduling Heuristics for Gird Systems with Varying Resource Availability", International Conference on Parallel and Distributed Computing Systems, vol. 10, no. 3, pp. 622-627, November 2006.

[6] P.Dabas, A. Arya, "Grid Computing: An Introduction of distributed system", International Journal of Computer Science and Information Technologies, vol. 3,no. 3, pp.466-470, March 2013.

[7] Q. Zheng, "Dynamic Adaptation of DAGs with Uncertain Execution Times in Heterogeneous Computing Systems", IPDPS Workshops IEEE, vol.60, no.6, pp. 1-8, April 2010.

[8] S. Gavaskar, D.V.Subbarao, "A survey of distributed fault tolerance strategies", International Journal of Advanced Research in Computer and Communication Engineering, vol. 2,no. 11, pp.4323-4327, November 2013.

[9] T. Kokilavani, G. Amalarethinam, "Load Balanced Min-Min Algorithm for Static Meta-Task Scheduling in Grid Computing", International Journal of Computer Applications, vol. 20,no. 2, pp. 43-49, April 2011.

International journal Of Business Management
ISSN NO. 2349-3402
Available at www.ijbm.co.in
VOL. 3 Issue 1, 2017

[10] T.Altameem, "Fault Tolerance Techniques in Grid Computing Systems",International Journal of Computer Science and Information Technologies, vol. 4, no.6,pp.558-568, May 2013.

[11] B. Yagoubi, Y.Slimani, "Task Load Balancing Strategy for Grid Computing",Journal of Computer Science, vol. 3, no. 3, pp. 186-194, June 2007.

[12] J.C Aswal, M.S Pal, O.P Gupta, "Load balancing strategies for Grid computing", Electronics Computer Technology (ICECT), 2011 3rd International Conference on , vol.3, no.8, pp. 239-243, April 2011.

[13] L.Yawei, L.Zhiling, "A Survey of Load Balancing in Distributed Computing", vol. 3314, no. 10, pp. 280-285, April 2005.

[14] U.K Kumar, "A Dynamic Load Balancing Algorithm in Computational Mobile Using Fair Scheduling", JCSI International Journal of Computer Science, vol. 8, no. 5, pp. 123-129, September 2011.

[15] R. Garg, A.K. Singh, "Fault Tolerance in Distributed computing: state of the art and open issues", vol. 2, no. 1, pp. 88-97, Feburary 2011.

[16] S. Gambhir , S. Goyal, "Reliable Task Allocation in Distributed Mobile Computing System with random node movement: Replication and Load Sharing Approach", International Journal of Advanced Research in Electronics and Communication Engineering, vol.3, no. 6, pp. 112-118, June 2014.

[17] P. K. Yadav, M. P. Singh, K. Sharma, "An Optimal Task Allocation Model for System Cost Analysis in Heterogeneous Distributed Computing Systems", International Journal of Computer Applications, vol. 28, no. 4, pp. 30-37, August 2011.