International Journal of Contemporary Research in Computer Science and Technology (IJCRCST) ISSN: 2395-5325
Volume 3, Special Issue 3 (September '2017)
Proceeding of - International Conference on Recent Trends in Computational Life Science and Information Technology,
Conference held at Tiruppur Kumaran College for Women,Tirupur, Tamilnadu,India.

VIRTUALIZED RESOURCE SCHEDULING FOR LOAD BALANCING AND FAULT TOLERANCE IN MULTICLUSTER GRID ENVIRONMENT

V.Indhumathi,
Research Scholar,
Department of Computer science,
Periyar University,
Salem, Tamilnadu, India.

Dr.G.M.Nasira,
Head of the Department,
Department of Computer Science,
Chikkanna Govt.Arts College,
Tirupur, Tamilnadu, India.

Abstract: In the current situations, complex real time applications depend on decentralized location for handling many practical problems. Therefore distributed clusters are used with unlimited computing resources. Different computers are interconnected through high-speed networks to provide efficient high performance computing platform for solving complex real time applications such as Heterogeneous clusters. In such environment, the constraints of distributed systems can be switched to develop the underutilized computing resources in all regions throughout the world for dispersed jobs. Even though Resource Scheduling, Load balancing and fault tolerance is well known research field, there are lots of new challenges in grid environments such as receiving up normal arrival of jobs, clustering jobs to resources, tolerate fault during load balancing are the main consideration of our proposed Benders decomposition.

Keywords: Grid environment, Heterogeneous clusters, High performance computing, distributed systems, Resource scheduling, Load balancing , Fault tolerance, complex real time applications, decentralized location and Benders decomposition.

Notes: This article is a revised and expanded version of a paper "Improved Fault Tolerant in Workload Execution through Quality Particle Swarm Optimization for Grid Environment"presented at[INDIACOM -2016, BVICAM,Newdelhi,17/3/2016] and "Service Oriented Architecture for Load Balancing with Fault tolerant in Grid Computing" presented at[ICACA-2016, Bharathiar University, 26/10/2016] and this article is extended version of the paper entitled "Resource Monitoring and Prediction with Fault Tolerance in Grid Environment through Meta heuristics" published in [IJAER,RIP, Dec 2015].

I.INTRODUCTION

A grid is a distributed computational and storage environment often composed of heterogeneous autonomously managed subsystems. Resource allocation, fault tolerance and job Scheduling are the core functions of grid computing. Load Balancing and fault tolerance techniques removes the limitations that exist in traditional shared computing environment by acquiring the adequate information of available resources, and which has becomes a great importance in the distributed computing system. Mostly problems arise due to dynamic sharing of resources which is termed as resource contention. A computer network, often simply referred to as a network, is a collection of computers and devices interconnected by communications channels that facilitate communications among users and allows users to share resources[1][2]. Among distributed environments, such as clusters, complexity of grids mainly originates from decentralized management and resource heterogeneity [3]. To be able to fully benefit from such grid systems, resource management and scheduling are key grid services, where issues of task allocation, load balancing and fault tolerance represent a common challenge for most grids. Grid computing is broadly into two categories compute and data grids. In compute grids, the main resource that is being managed by the resource management system is compute cycles (i.e., processors); while in data grids the focus is to manage data distributed over geographical locations. The type of grid system it is deployed in affects the architecture and the services provided by the resource management system.

Grid computing is a special type of parallel computing that relies on complete computers (with onboard CPUs, storage, power supplies, network interfaces, etc.) connected to a network (private, public or the Internet) by a conventional network interface, such as Ethernet. Grid computing environment is a distributed, shared environment implemented via the deployment of a persistent, service infrastructure that supports the creation and resource sharing within Distributed communities [6]. In addition to timeliness requirements, quality of service (QoS) requirements must be addressed in various hard and soft real-time systems. In order to improve the stability of the system, each task selects different QoS levels by varying its period or execution time. Noticeably, the aforementioned QoS-aware real-time systems must then incorporate inherent high reliability features. Growing evidence shows that scheduling is an efficient approach to achieving high performance of applications in parallel systems like clusters**.**

II.MOTIVATION AND NEED FOR RESEARCH

Due to varying resource availability and resource failure becomes commonplace, often resulting in loss in throughput values, response time and delay of executing jobs. These characteristics often lead to strong variations in grid

availability, which in particular depends on resource and network failure rates, administrative policies, and fluctuations in system load. Apparently, runtime changes in system availability can significantly affect application (job) execution. Since for a large group of time-critical or time consuming jobs delay and loss are not acceptable.

The motivation of the research is to consider fault tolerant scheduling and balancing application of load for a computational grid by taking into account grid architecture, computer heterogeneity, resource unpredictability and communication delay. Providing fault tolerance in a distributed environment, while optimizing resource utilization and job execution times, is a considered as motivation of the research. Through stronger consideration of fault tolerant scheduling and balancing application load for a computational grid by taking into account grid architecture, computer heterogeneity, resource unpredictability, and communication delay will lead effective management of resource in terms of prolonging the life time.

III. CONTRIBUTION TO THE RESEARCH

Mostly problems arise due to dynamic sharing of resources which is termed as resource contention. One of the challenging problems is deciding the destination nodes where the tasks of grid application are to be executed in consideration with delay and make span of the task queries. The load balancing is benefited from the collocation process which potentially leads to high resource utilization and lower queue wait time by allowing parallel jobs to run when need more processor than available.
The major contributions of this research are summarized as follows:

- Our first Contribution Resource Aware Fault Tolerance Scheduling (RAFTS) using Metaheuristic conditions in the Heterogonous Environment for real-time tasks running on clusters in general and heterogeneous clusters specifically. One of the challenging problems is deciding the destination nodes where the tasks of grid application are to be executed through Meta heuristics.
- Our second Contribution Fault Management Framework (FMF) in Service oriented architecture is a novel approach is balancing the load and resource fault by modeling autonomous services. This technique is a fault-tolerant version of the first technique. Our approach eliminates the complexity of a site to gather current state information of the whole grid since real time monitoring of whole grid will cause system overhead and is completely unrealistic in large scale grid environment. Fault Management Framework attempt to pick up the response time of user's proposed applications by ensures maximal utilization of available resources.
- Our third Contribution Heuristics Swarm Optimization (HSO) observes changes happening on the System work load plus reallocates the work consequently with fault tolerant. This algorithm usually composed of three strategies: relocate strategy, location strategy and information strategy**.** Relocate strategy make a decision on which tasks be suitable for move to further nodes for processing. Location strategy recommends a isolated node to accomplish a moved task. Information policy is the information hub used for load balancing algorithm**.**
- Our fourth Contribution Quality Particle Swarm Optimization (QPSO) includes two types of PSO for calculating the site and speed of task such as continuous version and binary version. Hyper-parameter selection is a kind of continuous optimization problem. Feature selection is a kind of binary optimization problem. We have to propose an optimization algorithm which hybridizes continuous PSO and binary PSO together, namely QPSO. A system is initialized with a number of random tasks and searches a multi-dimensional solution space for optima by updating task generations. Each task calculates its own speed and updates its location in each iteration until the termination condition is met to classify the task into resources.
- Our Fifth Contribution Load Balanced Fault Tolerant Architecture (LBFT) is a new dynamic load balancing algorithm for reduce the implementation time of applications in computational grid.

IV. RELATED WORK AND ANALYSIS

A wide variety of scheduling algorithms have been developed to provide fault tolerance for clusters supporting real-time applications. To support creation of nimble applications for computational grids, J.S. Vetter and D.A. Reed (2000) **'**Real-Time Performance Monitoring, Adaptive Control, and Interactive Steering of Computational Grids' **[7]** believe one must eliminate the barrier that separates program creation from execution and post-mortem optimization. This literature based on a suite of performance instrumentation, analysis, and presentation tools that include distributed performance sensors and resource policy actuators, fuzzy logic rule bases for adaptive control, and immersive visualization systems for real-time visualization and direct manipulation of software behavior. RPS (Resource Prediction System) is a publicly available toolkit that allows a practitioner to straightforwardly create flexible online and offline resource prediction systems in which resources are represented by independent, periodically sampled, scalar-valued measurement streams have been studied via P.A. Dinda (2006) 'Design, implementation, and performance of an extensible toolkit for resource prediction in distributed systems'[11].The systems predict the future values of such streams from past values and are composed at runtime out of a large and extensible set of communicating components that are in turn constructed using RPS's extensible sensor, prediction, wavelet, and communication libraries.

In the past few years, researchers have proposed scheduling algorithms for parallel system. However, the problem of grid scheduling is still more complex than the proposed solutions. Therefore, this issue attracts the interests of the large number of researchers. Current systems of grid resource management was surveyed and analyzed based on classification of scheduler organization, system status, scheduling and rescheduling policies. However, the characteristics and various techniques of the existing grid scheduling algorithms are still complex particularly with extra added components. T. Xie and X. Qin (2006)' Scheduling Security-Critical Real-Time Applications on Clusters '[12] proposed

security overhead model that can be used to reasonably measure security overheads incurred by the security-critical tasks. At the present time, job scheduling on grid computing is not only aims to find an optimal resource to improve the overall system performance but also to utilize the existing resources more efficiently. F. Harada, T. Ushio, and Y. Nakamoto (2007) 'Adaptive Resource Allocation Control for Fair QoS Managementr'[13] , resource utilization is assigned to each task through an online search for the fair QoS level based on the errors between the current QoS levels and their average Recently, many researchers have been studied several works on job scheduling on grid environment. Some of those are the popular heuristic algorithms, which have been developed, are min-min, the fast greedy, tabu search and an Ant System. The heuristic algorithms proposed for job scheduling in and rely on static environment and the expected value of execution times.

A load balancing algorithm aims to increase the utilization of resources with light load or idle resources thereby freeing the resources with heavy load. The algorithm tries to distribute the load among all the available resources. At the same time, it aims to minimize the makespan with the effective utilization of resources. S. Ghosh, R. Melhem, and D. Mosse(1997) 'Fault-Tolerance Through Scheduling of Aperiodic Tasks in Hard Real-Time Multiprocessor Systems '[14] schedule multiple copies of dynamic, aperiodic, nonpreemptive tasks in the system, and use two techniques that we call deallocation and overloading to achieve high acceptance ratio. R.U. Payli, E. Yilmaz, A. Ecer, H.U. Akay and S. Chien.(2004) 'DLB: A Dynamic Load balancing tool for Grid Computing' [15]. The approach for AWLB of parallel applications on heterogeneous resources has been scheduled based on optimal distributions. The AWLB provides an optimal distribution of the divisible workload between participating processors according to the computing environment characteristics and the application requirements. The main generic parameters that define a parallel application performance are the application parameter, where is the total amount of application communications, i.e., data to be exchanged (measured in bit) and is the total amount of computations to be performed (measured in Flop); The resource parameters, where is the available performance of the i^{th} processor (measured in Flop per second) and is the network bandwidth to this node (measured in bits per second). The AWLB algorithm is based on the benchmarking of the available resources capacity, defined as a set of individual resource parameters, and experimental estimation of the application parameter.

V.CHALLENGES IN RESOURCE SCHEDULING WITH FAULT TOLERANCE

Hybrid fault tolerant load balancing method merges the merits of active and passive replication method for fault tolerance which includes performance driven load balancing. Adaptive Scheduling Algorithm (ASA) overcomes the problems in grid architecture, resource and job heterogeneity, communication delay and resource unpredictability for continuous job allocation [17].

The following problem addressed in this study is outlined as complexities in Grid environment resource scheduling are:

- Size (large number of nodes, providers, consumers)
- Heterogeneity of resources (PCs, Workstations, clusters, and supercomputers, instruments, databases, software)
- Heterogeneity of application requirements (CPU, I/O, memory, and/or network intensive)
- Heterogeneity in resource demand patterns (peak, off-peak, ...)
- Applications need different QoS at different times (time critical results). The utility of experimental results varies from time to time.
- Geographical distribution of users & located different time zones.
-

VI. AIM AND OBJECTIVE OF THE PROPOSED RESEARCH

In this research work, we have suggested a method for effective load balancing and fault tolerance in the large scale workloads with varying characteristics by using Distributed Virtualized Scheduling framework is Benders decomposition algorithm. Meta heuristics conditions are used to allocate task to resources. Our experimental results indicate that the efficiency and accuracy of our system meet the demand of online system for grid resource monitoring and prediction.

VII.PROPOSED BENDERS ECOMPOSITION ALGORITHM

The proposed work has been implemented and tested using Grid simulator. The distributed resources and coupling services like Resource information and process information of task for monitoring and prediction of resource scheduling is diagrammatically shown in figure 1.

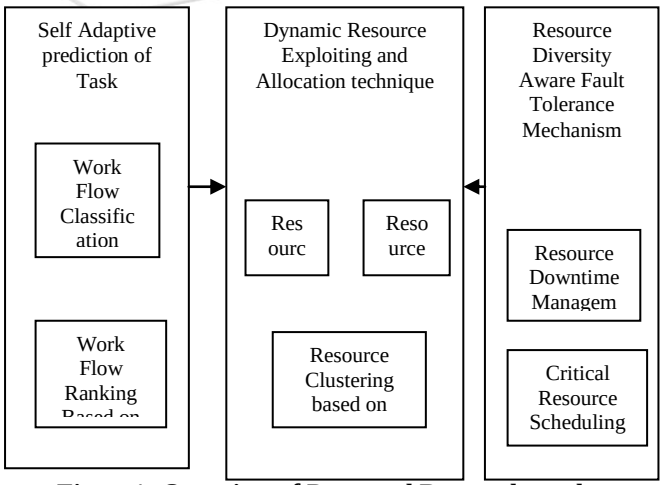

Figure1: Overview of Proposed Research work

Figure 1 illustrates the flow process of the proposed methodologies for providing the reliable resource schduling with fault tolerance for job assignments that can achieve the minimum response time, maximum resource utilization and a well balanced load across all the computing resources involved in a grid.

A. RESOURCE AWARE FAULT TOLERANCE SCHEDULING METHOD

The design of Resource aware fault tolerance method used to build a distributed system for grid resource monitoring and prediction. Major achievements includes the design and evaluation of system architecture for grid resource monitoring and prediction through Meta heuristic conditions. We discuss the key issues for system implementation, including machine learning based methodologies for modelling and optimization of resource prediction models in terms of handling fault tolerance. Our experimental results indicate that the efficiency and accuracy of our system meet the demand of online system for grid resource monitoring and prediction.

➢ Self Adaptive Prediction Technique (SAPT) for Workflow Computation

A self adaptive prediction technique for workflow computation used to classify and cluster the workflow or task into different categories based on different parameters. The sample set is separated into three parts: training set, validation set, and test set. The learning algorithm with a sample of training set (i.e text,pdf and image) is feeded, and one by one is repeated until all samples are used. For some algorithms, the training procedure runs only once; for others, iterations are needed. The trained models are feeded with all samples of validation set, and the errors all recorded between true data and predicted ones. Grid user logs on information node and customizes three terms before sending a monitoring/prediction request: which node, which resource type, and how long the prediction will last. A prediction request is then created accordingly and sent to information service. Information service launches the monitoring work flow by sending a monitoring request to the monitoring service.

Algorithm: Workflow Classification using Self Adaptive Prediction Technique

1. Initialize the population to the candidate solutions
2. Randomly select the candidate solutions from the candidate pool to the individual in the population
3. While Fitness criteria is not reached

Update the population of the solution through indexing & ranking using Reproduction operator

Applying Taylor series for Evolution in the candidate pool or population set using

$$p^x = 1 + \frac{x}{n(n-l)} + \frac{x^2}{n(n-2)} + \frac{x^3}{n(n-3)} + \cdots,$$
$$0 < x < n-1$$

Where 0 indicates the initial task or candidate solutions in the candidate pool

n-1 indicates the final task or Candidate solution in the candidate pool

p^x is the permutation set of the task and Resources

➢ Dynamic Resource Exploiting and Allocation(DREA) technique for Resource utilization

A dynamic Resource Exploiting and Allocation Technique for resource utilization is utilized for workflow to grid system model, and its default hyper parameters are set. The sample set is separated into three parts: training set, validation set, and test set. The learning algorithm with a sample of training set is feeded, and it one by one repeated until all samples are used.

The training data set is given by

$$D = \{(x_1, y_1)\Lambda(x_i, y_i)\Lambda(x_l, y_l)\},$$
$$x \in R^n, y \in \{-1,1\} \quad \text{----- (1)}$$

Where, l – number of training data,
X_i – Training data,
y_i– class label as 1 or -1 for x_i.

A nonlinear function is adopted to map the original input space R^n into N-dimensional feature space.

$$\psi(x) = \varphi_1(x), \varphi_2(x), ..., \varphi_N(x) \text{ ----- (2)}$$

The separating hyper plane is developed in this N-dimensional feature space. Then the classifier is represented as,

$$y(x) = \text{sgn}(\omega.\psi(x) + b) \text{ -- (3)}$$

Where ω - weight vector and b- scalar.

In order to obtain the optimal classifier, $\|\omega\|$ should be minimized subjected to the following constraints

$$y_i[\varphi(x_i).\omega + b] \geq 1 - \xi_i, \text{ i= 1, 2… l. ----- (4)}$$

The variable ξ_i is the positive slack variables, necessary for misclassification.

For some algorithms, the training procedure runs only once; for others, iterations are needed. The trained model is feeded with all samples of validation set, and the errors are recorded between true data and predicted ones. A resource sensor is activated as requested, and timely updated monitoring data are then sent back to information service through local and group registers; historical records are stored in the database for achieving prediction.

➢ Resource Diversity Aware Fault Tolerance (RDAFT) Mechanism

Metaheuristic is considered to be fault handling mechanism as it may make few assumptions about the optimization problem being solved, and so they may be usable for variety of the conditions in resource and task scheduling in the grid computing [8][9]. Meta heuristics works on stochastic optimization Method to generalize the deterministic case for deterministic heterogeneous resource types in the grid. Deterministic case of the resource variants depends upon the approximation methods.

➢ Algorithm for the Resource Diversity Aware Fault Tolerance

Create a Heuristic model to be connected by the job
Bind the Resource to an empty Communication using optimization technique $\sum\mu_i+\sum\mu_r$
Where I = task, R = resource
Listen to this port utilizing $\sum\mu_i$
BEGIN
While condition = TRUE do

If a new connection request arrives then
The optimization server accepts the connection from the Work load $\sum \mu_r$
Process id is computed as $K_{p=1}=1+\alpha\ (4+\beta)$
If process Id = Meta information of Work ID
Then
Back to listening to the designated port
Else {in the child process}
Child: close the listening port
Child: receive the request information from the Task
Child: perform sampling transfers $\alpha\ (4+\beta)$
Child: build a mathematical model and process the sampling results α
Child: send back the optimized parameters to the Workload β
Child: close the connection
Child: terminate
End if
Else {no new connection request comes}
END

B. FAULT MANAGEMENT FRAMEWORK

In order to accomplish the user prospect in terms of performance and competence, the Grid system desires Fault Management Framework for the sharing of tasks with fault tolerance. In this implementation, may be the running service occurrences are at risk to failure earlier than completion of their tasks because of unpredicted resources such as RAM or disk swap. This would lead to introduce Fault Management Framework to avoid the above said problem. The attitude of the Fault Management Framework is to give error handling that is peripheral to SOA. The structure is engaged through guidelines distinct in XML. These guidelines are reusable over components plus preserve to hold runtime faults. On one occasion a fault is caught, the procedure describes events that can be utilized for the SOA instance such as resource failure, retry, human intervention, rethrow fault, abort, and etc.

➢ **Dynamic Workload distribution**

In this section a dynamic workload distribution called, Dynamic Resource Scheduling (DRS) is presented to enhance the performance level on workload distribution. The DRS follows the fault management framework in SOA is the structure engaged through guidelines distinct in XML. These guidelines are reusable over components plus preserve to hold runtime faults. On one occasion a fault is caught, the procedure describes events that can be utilized for the SOA instance such as resource failure, retry, human intervention, rethrow fault, abort, and etc.

// Workload distribution using Dynamic Resource Scheduling
Step 1**: Loop**
Step 2: Wait for service request from User, Collect all services as
Step 3: Set $S = \{S_1, S_2, \ldots\ldots, S_\infty\}$ (5)
Step 4: Apply Quick sort to the set S, it divide the service to all nodes
Step 5: Calculate average load of the service as
$Avg(S) = \frac{\sum_{n=1}^{\infty} S_n}{n}$ (6)
Step 6: Identify fault of each service
Step 7: **If** (activity occurs) **then**
Step 8: Call fault policy, it catch the fault and take necessary action
Step 9: **Else** task is executed based on workload
Step 10: Iterates over all services offered by user
Step 11: Return the results to user
Step 12: **End Loop**

C. DESIGN OF HEURISTICS SWARM OPTIMIZATION

In order to formulate computational grids supplementary valuable and reliable fault tolerant system is essential. In favor of that we intend HSO (Heuristics Swarm Optimization) Algorithm to observe changes happening on the system work load plus reallocate the work consequently with fault tolerant. This algorithm usually composed of three strategies: relocate strategy, location strategy and information strategy**.** Relocate strategy make a decision on which tasks be suitable for move to further nodes for processing. Location strategy recommends a isolated node to accomplish a moved task. Information policy is the information hub used for load balancing algorithm**.**

➢ **Random selection of Resources**

All resource b_i uphold Q amount of adjoining resources QNSet, in which the grid scheduler be able to exploit to choose adjoining resources for off-load tasks. A neighbor meant for every resource is fashioned in requisites of transfer delay. For b_i,b_k is measured as its adjoining resource as long as the transfer delay flanked by b_i and b_k is inside β times of the transfer delay between b_i and the adjacent resource and load of b_k is less than b_i. For all resource, other adjacent resources are sort out by transfer delay in ascending order. After this process is more, the primary level resource is chosen as the nearest resource.

$$\beta = \frac{\sum_{i=1}^{N} Td_{ik}}{\sum_{i=1}^{N} Td_{inearest(b_i)}} \quad (7)$$

Where Td_{ik} indicate the transfer delay from resource bi and b_k. $Td_{inearest}$ indicate the transfer delay from the nearest resource of resource b_i to itself. Every resource b_i also uphold M quantity of associate resources QPSet, which the Grid scheduler be capable of utilize to decide on a associate resource for off-load tasks. For a resource b_i, a resource b_k is measured as its associate resources as long as the processing ability of b_k is greater than or equal to the processing ability of b_i and load of b_k is less than b_i. For each resource, other partner resources are sort out by the processing power in descending order. When this process is over, the first ranked resource is chosen as the fastest resource.

// Workload distribution algorithm
Step 1: **WHILE** balance is not good enough
Step 2: **FOR** each node in the graph
Step 2.1: Compute its *energy* (computation/communication requirements)
Step 2.2: **IF** probabilistic computation designate that node must travel **then**
Step 2.2.1: Select a set of *neighbors* allotted to further processors
Step 2.2.2: ***FOR*** *every neighbor* in the set
Step 2.2.3:Calculate impact of reassigning the node

Step 2.2.4: Shift the node to the finest neighbor panel
Step 2.3 : **End if**
Step 3: **End for**

// Algorithm for Ranking the task

Step 1: Calculate rank value to give priority to sort tasks to be executed
Step 2**: for** all task estimate average implementation time on every processors
Step 3: **if** task is final one
Step 3.1: status of task = average of the task
Step 3.2: **else** status of task = average of task + max(rank(predecessors)) + weight of channel
Step 4: **end if**
Step 5:**end for**

D.QUALITY PARTICLE SWARM OPTIMIZATION

In this paper, propose a outline for grid resource monitoring and prediction using Application aware provisioning mechanism in heterogeneous distributed systems. The user's tasks negotiating with resource providers based on their essential Quality of Service and on the equivalent price to reach a Service Level Agreement using algorithm Quality Particle Swarm Optimization (QPSO). Hyper-parameter selection is a kind of continuous optimization problem. Feature selection is a kind of binary optimization problem. We have to propose an optimization algorithm which hybridizes continuous PSO and binary PSO together, namely QPSO. A system is initialized with a number of random tasks and searches a multi-dimensional solution space for optima by updating task generations. Each task calculates its own speed and updates its location in each iteration until the termination condition is met to classify the task into resources.

- **Dynamic user priority**

The designing of Dynamic user priority compute an energetic user priority for individual users or for a grouping, depending on how the shares are assigned. The priority is active because it modify as soon as any changeable in formula is revised. By evasion a user's active priority steadily reduces behind a job starts, and the active priority instantly increases when the job finishes.

It computes the active priority for each user based on:

- The number of share allocate to the user
- The resources worn by jobs be in the right place to the user:
  - Number of job period reserved and in use
  - Run time of running jobs
  - Enlarge actual CPU time (not normalized). so that lately worn CPU time is prejudiced more deeply than CPU time used in the far-away past
  -

Dynamic Priority formula is,

Dynamic Priority = Number Shares / (CpuTime * CpuTimeFactor + Runtime * RunTimeFactor + (1 + Job Slots)

(8)

// Algorithm for dynamic user priority

Step 1: Fix a prediction model of Machine Learning algorithm
Step 1.1: Divide the trial set as Training set, Validation set and test set
Step 2: Using Learning algorithm with a training set; do again it one by one until all Models are used.
Step 3: With the learning algorithm, use validation set to find out the faults.
Step 3.1: When the faults are occurred possible prediction model is realized, and after that tested via test set.
Step 4: Use feasibility algorithm to develop hyper-parameters for prediction of tasks to resources for enhanced robustness (performance).

E.LOAD BALANCED FAULT TOLERANCE ARCHITECTURE

The designing of Load Balanced Fault Tolerance architecture (LBFT) focus on a new dynamic load complementary algorithm beside among fault tolerant scheduling approach through which successful load balancing and fault tolerance have been accomplished. Load connected through demanding service action be able to force a huge deal of runtime status of organization responsibilities ahead resource monitoring and newly concerned services, in this manner falling their scalability. As overhaul implementation consumes total sources plus depict near restrictions among agreed mechanism, a typical result be virtualizes the overhaul via arrange several instances of a overhaul occurrence diagonally various machines and transmit service requirements to all of the examined illustration through a task allocator.

LBFT architecture contains four components: The Load Balancing Interceptor, Dispatching Strategies, Ft Stateful services and MDM Data Hub. The Load Balancing Interceptor asks the Dispatching Strategy to which service the request message should be sent. Then the request sent to Fault Tolerant stateful services. The FT services first refer the Master Database Management (MDM) for which services are available, based on the request sent to that service. After executing the request, the result sent to FT stateful services. If any fault identified during execution the services immediately contact the FT services, then it immediately contact Fault Tolerant Collection (FT). The FT having list of faults and their solutions.

// Scheduled Load balancing algorithm
Step 1: **Begin**
Step 2: **If**(SD of load < SD of Threshold>
Step 3: **If**(Load > average load value)
Step 3.1: Add new node to Heavyloadedlist
Step 3.2: End if;
Step 4: **Else** (Load > threshold of heavy load value>
Step 4.1: Add new node to list;
Step 5: **Else if**(Load < threshold of light load value)
Step 6: **End**

International Journal of Contemporary Research in Computer Science and Technology (IJCRCST) ISSN: 2395-5325
Volume 3, Special Issue 3 (September '2017)
Proceeding of - International Conference on Recent Trends in Computational Life Science and Information Technology,
Conference held at Tiruppur Kumaran College for Women,Tirupur, Tamilnadu,India.

VIII.SIMULATIONS AND PERFORMANCE METRIC ANALYSIS

The proposed methods such as RAFTS method, FMF method, HSO method, QPSO method and LBFT method are implemented using Gridsim. The performance of proposed methods are evaluated with the metrics such as execution time, memory utilization, CPU utilization, cost and security (robustness).

A.Impact of Execution time

For a small number of clusters, the global amount of jobs move is high. This is outstanding to congestion groups which origin to describe the inter-clusters load balancing entailed a task transport between clusters. By raising the quantity of groups, the inclusive amount of jobs reassignment reduces considerably, encouraging a confined remove (intra-cluster), which strengthen our goal. Figure 2 demonstrate the difference of proposed approach achieved and those attained near the structured come up for 8 groups the same as the subjective number of tasks.

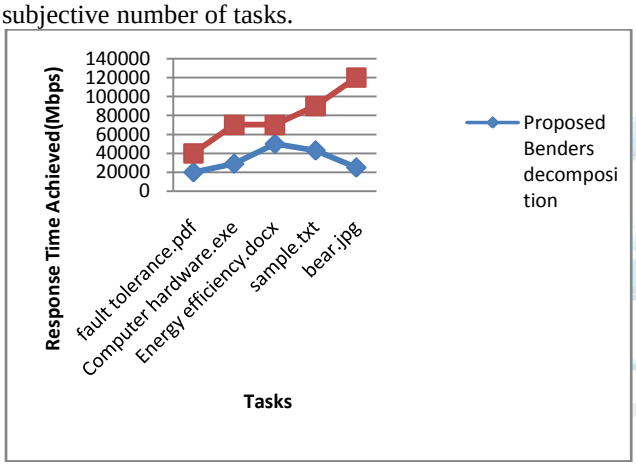

Figure 2: Measure of Response time

As shown in figure, RAFTS method provides better performance as compared to other existing method namely, PSO. This is because of the application of Dynamic Resource Exploiting and Allocation (DRE) in Resource Aware Fault Tolerance Scheduling method. With the support of Dynamic Resource Exploiting and Allocation, RAFTS method easily decreases the execution time while scheduling the tasks to resources which in turn decreases the execution time by 8% as compared to PSO.

B. Impact of Memory Utilization, CPU utilization and Cost

Benders decomposition algorithm is used to measure the memory utilization, CPU utilization, cost and robustness by divides the given task into more number of partitions based on the task size i.e. in our simulation we have to consider the tasks having greater than 1 Mb for decomposition. It scheduled each and every task located in resources to get minimum response time by using the f(x) value of each task, where f(x) is criteria condition to determine task which gets minimum response time. Depends upon the criteria condition algorithm divide the task into more than one. Make convergence after scheduling the task to resources. The Benders Decomposition algorithm is appraised for the above defined metrics. The results are compared with Particle swarm Optimization (PSO) algorithm. The performance comparison between Benders and PSO and the results of algorithms such as PSO and Benders are shown in figure 3, figure 4 and figure 5 respectively.

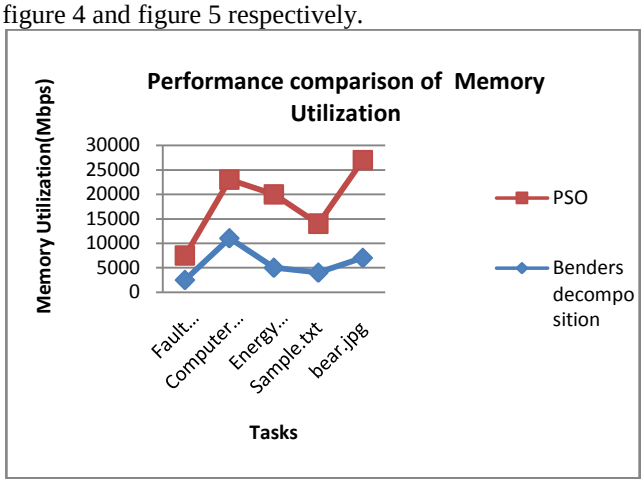

Figure 3: Measure of Memory utilization

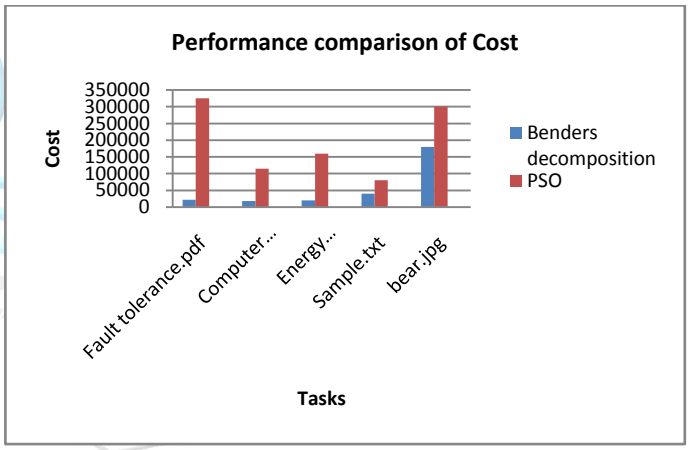

Figure 4: Measure of cost

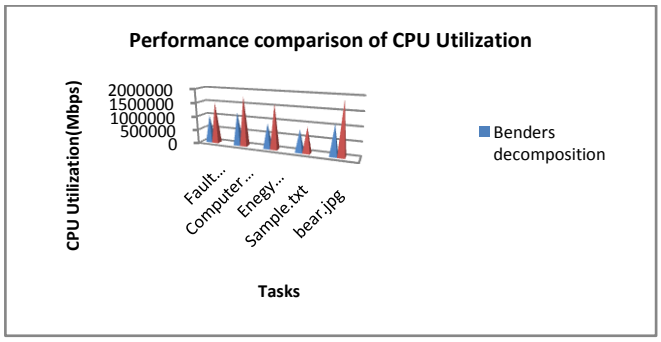

Figure 5: Measure of CPU utilization

Metaheuristics may make few assumptions about the optimization problem being solved, and so they may be usable for variety of the conditions in resource and task scheduling in the grid computing [8][9]. Meta heuristics works on stochastic optimization Method to generalize the deterministic case for deterministic heterogeneous resource types in the grid. Deterministic case of the resource variants depends upon the approximation methods. The main aim is to provide better performance through the factors of Execution

time, Memory utilization, Cost and CPU utilization of Benders decomposition is compared with existing algorithms.

IX.CONCLUSION

Our proposed contribution Benders decomposition algorithm includes techniques such as RAFT, FMF, HSO, QPSO and LBFT. The main objective of these techniques is to arrive at the job assignments that can achieve the minimum response time, maximum resource utilization, robustness and a well balanced load across all the computing resources involved in a grid. Gridsim version 2.6 has been used to evaluate the performance of the proposed approaches. The simulation results show that considerable performance gains are achieved in terms of the job's individual response time and the average response time. In the future we are using the proposed strategy in networking to provide better results.

XREFERENCES

[1] F. Berman, R. Wolski, H. Casanova, W. Cirne, H. Dail, M. Faerman, S. Figueira, J. Hayes, G. Obertelli, J. Schopf, G. Shao, S. Smallen, N. Spring, A. Su, and D. Zagorodnov, "Adaptive computing on the grid using AppLeS," IEEE Trans. Parallel Distrib. Syst., vol. 14, no. 4, pp. 369–382, Apr. 2003.

[2] V. V. Krzhizhanovskaya and V. V. Korkhov, "Dynamic load balancing of black-box applications with a resource selection mechanism on heterogeneous resources of the grid," in Conf. Parallel Comput. Technol. (PaCT), LNCS, Berlin/Heidelberg, 2007, vol. 4671, pp. 245–260.

[3] H.Y. Chang, K.C. Huang, C.Y. Shen, S.C. Tcheng, and C.Y. Chou, "Parallel Computation of a Weather Model in a Cluster Environment," J. Computer-Aided Civil and Infrastructure Eng., vol. 16, no. 5, pp. 365-373, Sept. 2001.

[4] T. Xie and X. Qin, "Scheduling Security-Critical Real-Time Applications on Clusters," IEEE Trans. Computers, vol. 55, no. 7, pp. 864-879, July 2006.

[5] C.M. Krishna and K.G. Shin, Real-Time Systems. McGraw-Hill, 2001.

[6] T.F. Atdelzater, E.M. Atkins, and K.G. Shin, "QoS Negotiation in Real-Time Systems and Its Application to Automated Flight Control," IEEE Trans. Computers, vol. 49, no. 11, pp. 1170-1183, Nov. 2000.

[7] J.S. Vetter and D.A. Reed, "Real-Time Performance Monitoring, Adaptive Control, and Interactive Steering of Computational Grids," Int'l J. High Performance Computing Applications, vol. 14, no. 4, pp. 357-366, 2000.

[8] D.M. Swany and R. Wolski, "Multivariate Resource Performance Forecasting in the Network Weather Service," Proc. ACM/IEEE Conf. Supercomputing, pp. 1-10, Nov. 2002.

[9] P.A. Dinda and D.R. O'Hallaron, "Host Load Prediction Using Linear Models," Cluster Computing, vol. 3, no. 4, pp. 265-280, 2000.

[10] E. Caron, A. Chis, F. Desprez, and A. Su, "Design of Plug-in Schedulers for a GRIDRPC Environment," Future Generation Computer Systems, vol. 24, no. 1, pp. 46-57, 2008.

[11] P.A. Dinda, "Design, Implementation, and Performance of an Extensible Toolkit for Resource Prediction in Distributed Systems," IEEE Trans. Parallel and Distributed Systems, vol. 17, no. 2, pp. 160-173, Feb. 2006.

[12] T. Xie and X. Qin, "Scheduling Security-Critical Real-Time Applications on Clusters," IEEE Trans. Computers, vol. 55, no. 7, pp. 864-879, July 2006.

[13] F. Harada, T. Ushio, and Y. Nakamoto, "Adaptive Resource Allocation Control for Fair QoS Management," IEEE rans. Computers, vol. 56, no. 3, pp. 344-357, Mar. 2007.

[14] S. Ghosh, R. Melhem, and D. Mosse´, "Fault-Tolerance Through Scheduling of Aperiodic Tasks in Hard Real-Time Multiprocessor Systems," IEEE Trans. Parallel and Distributed Systems, vol. 8, no. 3, pp. 272-284, Mar. 1997.

[15] R.U. Payli, E. Yilmaz, A. Ecer, H.U. Akay and S. Chien. 'DLB: A Dynamic Load Balancing Tool for Grid Computing', Parallel CFD Conference, May 24-27, 2004, Grand Canaria, Canary Islands, Spain.

[16] L.T. Lee, D.F. Tao, and C. Tsao, "An Adaptive Scheme for Predicting the Usage of Grid Resources," Computing Computers and Electrical Eng., vol. 33, no. 1, pp. 1-11, 2007.

[17] Jasma Balasangameshwara "Improving Fault tolerant Load balancing Algorithms in Computational Grids" Information Engineering and Electronic Business, Vol 6,pp. 53-62,2015.

[18] V.Indhumathi,and G.M.Nasira , "Fault Tolerance in Job scheduling through Fault Management Framework Using SOA in Grid", ICTACT Journal on Soft Computing, Volume 7,Issue-2,Page No. 1381 – 1385, January 2017.

[19] V.Indhumathi "Improved Fault Tolerant in Workload execution Through Quality Particle Swarm Optimization for Grid Environment" International IEEE conference on Computing for sustainable Global Development, INDIACOM 2016 pp.5040 – 5045,Mar 2016.

[20] V.Indhumathi and Dr.G.M.Nasira, "Resource Monitoring and Prediction with Fault Tolerance in Grid Environment through Meta Heuristics", International Journal of Applied Engineering and Research, vol 10, no 24, pp43993-44000, Dec 2015.