

Host Load Prediction in Computational Grid Environment

Ankita Agrawal

(M.Tech)

Medicaps Institute of Technology & Management Indore (M.P) India

Rudesh Shah

ABSTRACT

When sudden load arises in a grid then load should be transferred to some idle node hence due to load sharing, server down condition not occur hence we predict the load on node and shared it to idle node this is termed as load forecasting. In this paper, we did simulation of grid CPU load in distributed manner which provide the monitoring on each host in network and load of grid is predicted using bpn (back propagation neural network) algorithm. Which provides the effective results in prediction, in addition of that a new predictive algorithm is proposed implemented and compared to the bpn algorithm.

In proposed algorithm we calculate the accuracy and compared with bpn algorithm accuracy, after implementation of both methods and simulation of Host load we find the proposed method is much effective then the previously proposed method of BPN algorithm.

Keywords— *bpn, grid computing.*

1. INTRODUCTION

Grid is a combination of computer resources from multiple administrative domains to reach a common goal .it is loosely coupled, heterogeneous and geographically dispersed. Grid can be used for a variety of different purposes Ex- educational institutions, offices, industries. Grid size can be vary according to our need, It gives a feeling of Desktop supercomputing means we are sitting in front of our desktop but we are connected to supercomputer. For achieving high performance in a grid computing environment Task scheduling and prediction of execution time of every applications is an important component. [1]

There are at least two pre-requisites for this situation. First, the request must be executable remotely and without unnecessary overhead. Second, remote machine must meet any special hardware, software, or resource requirements compulsory by the application. If the quantities of input and output are huge, more assumption and planning might be required to efficiently use the grid. It would typically not make sense to use a word processor remotely on a grid because there would probably be greater delays and more potential points of failure. A grid is used for a variety of purposes. Grids are frequently constructed with general-purpose grid middleware software collections or library. [2]

The allocation of the resources and the scheduling of tasks are basic problems in grid computing, and there is no doubt that resource performance is the most influencing factor within such area. The performance information of grid resources are mainly achieved by means of two mechanisms: monitoring and prediction. Grid resource monitoring aims to acquire the status, distribution, load as well as the fault situation of the resources in grid environment by means of monitoring methods. While grid resource prediction aims to handle the variation principles and running traces of grid resources by means of modelling and analysing on historical monitoring data. In a word, monitoring can provide historical information and the current information, while prediction provides future variation information. These two mechanisms are supplement to each other. Grid resource monitoring and prediction are inevitable in grid computing system. The grid needs a large amount of monitoring and prediction data: [3]

- To carry on performance analysis, service control, bottleneck elimination and fault diagnosis;
- to provide reliable direction for grid resource allocation, job scheduling as well as dynamic load balancing;
- To help grid users to finish computing tasks while minimizing cost on time, space, and money.

In some organizations, even the server machines can often be relatively idle. Grid computing provides a context for using these underutilized resources and thus has the possibility of significantly increasing the efficiency of resource usage.

The execution time of a CPU-bound task on a host is tightly related to the CPU load on the host. In fact, the relationship between the execution time of a CPU bound task and the measured load during the execution is almost linear for some application. [4]

If we are able to predict the load during the execution of a task on a host, execution time of the task on the host could be easily predicted. Therefore, for achieving high application performance and efficient resources use, host load prediction can be useful for guiding scheduling strategies. Because of resource contention host load and availability vary over time. Resource contention occurs because applications compete for resources with unknown workloads from other use and makes the load prediction problem more difficult.

2. BACKGROUND

In this paper [5] author presents and evaluate a new and innovative method to predict the one-step ahead CPU load in a grid. The prediction strategy forecasts the future CPU load based on the tendency in several past steps and in previous

similar patterns, and uses a polynomial fitting method. Experimental results demonstrate that new prediction strategy achieves average prediction errors that are between 37% and 86% lower than those incurred by the previously best tendency-based method. Predictions of future system performance can be used to improve resource selection and scheduling in a dynamic grid environment.

Some predictable factors such as trends, periodicity or scheduled events allow for proactive resource provisioning in order to meet fluctuations in workloads. However, proactive resource monitoring requires prediction models forecasting future workload patterns. This paper [6] proposes a multi-model prediction method, in which data are grouped into bins based on content locality, and an autoregressive prediction model is assigned to each locality preserving bin. The prediction models are shown to be identified and fitted in a computationally efficient way. Author demonstrates experimentally that our multi-model approach improves locality over the uni-model approach, while achieving efficient resource provisioning and preserving a high resource utilization and load balance.

The short-term modelling approach presented in this paper complements the long-term modelling and prediction from previous work. In the future, author plan to integrate these two approaches into a hierarchical framework [10], which should provide integrated prediction and modelling. Further, author plan to extend study to other traces. Finally, author plan to study the applicability of ideas in geographically distributed content delivery systems.

To achieve effective load balancing and a robust Grid environment, an ended load forecast for computational resources is increasingly required. Thus, this paper proposes [7] a method of predicting network and CPU load variance within a wide range, from several minutes to over than a week. This is the widest range of prediction of the existing algorithms in the load of computational resources for the Grid environment. The distinctiveness of algorithm is in using seasonal load variation for both load variance and one-step-ahead prediction. Seasonal fluctuation in CPU loads to network load variation especially for network load variance prediction. Moreover, it, the Markov model-based meta-predictor is used for one-step-ahead prediction, which is sensitive to late trends. Experiments demonstrate that algorithm gives a good curve for expected 8-day long load variance, and makes accurate one-step-ahead predictions. Mean error rate for one-step-ahead predictions is 9.4% in the case of network load, and 6.2% in the case of CPU load. Additionally, the least mean error rate for wider range forecasts is 5.5% for network load variation, and 3.6% for CPU load variation.

As future work, author plan to improve the precision, especially for CPU load variation, maximum values, and minimum values. author will, in addition, apply this method to PCs, workstations and so on, because we confirmed the effect of OUT method for router only use machines. Moreover, need to develop cooperation with the load-balancing system to improve practicability and determine the prediction format to provide forecast for other.

Author design a prediction method based on Bayes model to predict the mean load over a long-term time interval [8], as well as the mean load in consecutive future time intervals. Identify novel predictive features of host load that capture the belief, predictability, trends and patterns of host load. Author also determines the most effective combinations of these features for prediction. Method using detailed one-month traces of a Google data centres with thousands of computational units. Experiments show the

Bayes method provides high accuracy with a MSE of 0.0014. additionally, the Bayes method improves the load prediction accuracy by 5.6-50% compared to other state-of-the-art methods based on moving averages, autoregression, and/or noise filters.

For the selection and allocation of grid resources to current and future applications, grid job scheduling is playing a very vital role for computational grids. Author constitutes the building blocks for making grids available to the society. The efficient and effective scheduling policies, when assigning different jobs to specific resources, are very important for a grid to process high computing intensive applications. This paper presents [9] an agent based job scheduling algorithm for efficient and effective execution of user jobs. This paper also includes the comparative performance analysis of author proposed job scheduling algorithm along with other well-known job scheduling algorithms considering the quality of service parameters like waiting time, turnaround time, response time, total completion time, bounded slowdown time and stretch time. Author also conducted the QoS based evaluation of the scheduling algorithms on an experimental computational grid using real workload traces. Experimental evaluation confirms that proposed grid scheduling algorithms possess a high degree of optimality in performance, efficiency and scalability.

3. PROPOSED SYSTEM

Grids computing offer a way to solve Grand Challenge problems such as protein folding, financial modelling, earthquake simulation, and climate/weather modelling. Grids offer a way of using the information technology resources optimally inside an organization. But there are some problem arises when workloads under different grid nodes suddenly occur, and available resources are not sufficient for responding them in proper manner.

In addition of that the grid is affected by two major issue first resource scheduling and in seconds the fault occurrence. To monitor and evaluation of the network two techniques are frequently used statically analysis method and the predictive methods. In predictive methods system can forecast the upcoming load but most of the predictive methods are provide less accurate load during prediction thus a new algorithm is required to predict CPU load on different hosts in grid. Proposed solution includes the following work to design, develop and deploy successfully the complete the work.

1. Study of load problems on different application servers
2. Study of grid computing architecture and the current management schemes
3. Design architecture for grid load prediction
4. Tool implementation for load analysis based on above described parameters.
5. Prediction analysis under different performance parameters i.e. MAPE Error, accuracy and others
6. Identification and evaluation different load parameters.

In this proposed study includes the below given works are includes evaluating:

1. study of different strategies on host load monitoring and fault tolerances
2. identifying the solution steps
3. implement the most frequently used technique
4. propose a new technique to enhance the performance of previous work

5. evaluate the results and provides the comparative study

Our main aim to achieve in this paper is get or collect the load parameters from all the grid nodes organized to gather and according to these parametric values predict the future upcoming load over the server nodes. Additionally find the effect with increase the grid nodes in computational environment, the below given figure 1 shows the basic working of the system.

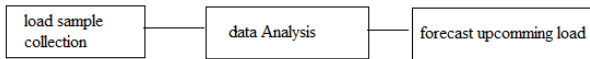


Fig 1: basic flow of system

Load sample collection from client machine is performed, these load samples are network and CPU consumption parameters for any particular time t1, this data is now updated to the server end to perform data analysis. After finding patterns over selected data system is capable to predict the network resource consumption over time t2.

For that purpose we work with two different algorithms first BPN and second self-pattern analysis algorithm designed by us.

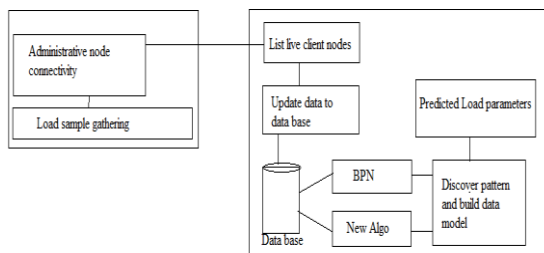


Fig2: system architecture

Overall system is divided into two main parts first server side and client side system. The description of diagram is given as

- **Load sample gathering:** here at the client node end application collect network speed, upload speed, downloads speed, CPU uses and other parameters
- **Administrative node connectivity:** that is a connectivity check at client end by which system test the connectivity for data transfer.
- **Data transfer to server:** if connectivity is available then client node sends data to server
- **List live client nodes:** as we know grid is collection of multiple high performance resource thus the currently connected and live resources are listed here for data sample collection.
- **Update data to database:** the collected samples are added to data base for analysis.
- **Database:** that is collection of data tables and other database elements here data is added in the form of data table.
- **BPN:** in this phase data is analyzed using back propagation neural network
- **New algo:** in this phase data is analyzed using our designed algorithm
- **Discover patterns and build data model:** according to the data algorithms make data model for negotiating for next arrived pattern.

- **Predict load parameters:** the predicted data is provided for administrative use.

4. IMPLEMENTATION& RESULTS

A. Back Propagation Algorithm

Artificial neural networks are composition of interconnecting artificial neurons these neurons are programmatically constructed, in computer science it is used to solving artificial intelligence problems. It is a data model created using mathematical complex calculations. The utility of artificial neural network models lies in the fact that they can be used to infer a function from observations and also to use it.

B. Components of Artificial Neural Network

1. A set of processing units;
2. An activation function;
3. Connections between the units where each unit is defined by a weight w_{ji} that measures the effect that the signal of unit j has on unit i ;
4. A propagation rule, which determines the effective input of the unit from its external inputs;
5. An external input for each unit;
6. A method for information gathering, learning rule;
7. An environment within which the system can operate, provide input signals and, if necessary, error signals.

The basic architecture of single-layer perceptron network, which include a single layer of output nodes; the inputs are fed directly to the outputs via a series of weights. In this way it can be considered the simplest kind of feed-forward network. Back propagation network consists of minimum three layers which is called multilayer perception. In this network an input layer, at least one middle hidden layer, and an output layer. Input units are connected in a feed-forward fashion with input units fully connected to units in the hidden layer and hidden units fully connected to units in the output layer. An input pattern is propagated forward to the output units through the intervening input-to-hidden and hidden-to-output weights when a Back Propagation network is cycled [10].

The basic back propagation algorithm includes three steps.

- The input pattern is given to the input layer of the network. These inputs are propagated through the network until they reach the output units. This forward pass produces the predicted output pattern.
- As it is a supervised learning algorithm, the preferred outputs are given as part of the training vector. The actual network outputs are subtracted from the desired outputs and an error signal is produced.
- This error signal is the basis for the back propagation step, where the errors are passed back through the neural network by computing the contribution of each hidden processing unit. Deriving the corresponding adjustment needed to produce the accurate output. The connection weights are then adjusted.

C. Algorithm Neural Network

The implementation of neural network is defined in two phases' one training and prediction: training process consumes data and designs the data model. Using this data model in next phase prediction of data is taking place.

Training:

1. Initialize two vectors one input and hidden unit and second output unit.
2. Here first is a two dimensional array W_{ij} is used and output is a one dimensional array Y_i .
3. Initial weights are random values put inside the vectors after that the we calculate the output as

$$x_j = \sum_{i=0} y_i W_{ij}$$

Where y_i is level of the j^{th} unit in the previous layer and W_{ij} is the weight of the connection between the i^{th} and the j^{th} unit.

4. Next, activity level of y_j is calculated by some function of the total weighted input.

$$y_i = \left[\frac{e^x - e^{-x}}{e^x + e^{-x}} \right]$$

When activity of the all output units have been calculated, the network computes the error E ,

$$E = \frac{1}{2} \sum_i (y_i - d_i)^2$$

Where y_i is activity level of the j^{th} unit in the top layer and d_i is the desired output of the j_i unit.

D. Calculation of error for the Back propagation algorithm is as follows:

- Compute Error Derivative (EA) is the difference between the actual and the desired activity:

$$EA_j = \frac{\partial E}{\partial y_j} = y_j - d_j$$

- Calculate the error changes as the total input received by an output changed

$$El_j = \frac{\partial E}{\partial X_j} = \frac{\partial E}{\partial y_j} \times \frac{dy_j}{dx_j} = EA_j y_j (1 - y_j)$$

- Calculate the error changes as a weight on the connection into an output unit is changed:

$$EW_{ij} = \frac{\partial E}{\partial W_{ij}} = \frac{\partial E}{\partial X_j} = \frac{\partial X_j}{\partial W_{ij}} = El_j y_i$$

- Calculate the overall effect on the error:

$$EA_i = \frac{\partial E}{\partial y_i} = \sum_j \frac{\partial E}{\partial X_j} \times \frac{\partial X_j}{\partial y_i} = \sum_j El_j W_{ij}$$

E. proposed algorithm

The proposed algorithm is an adoptive method of learning which learns also from the currently accepted data in database. Not much complex to understand and in the same time it is simple and easy to implement. The proposed adoptive algorithm steps are given using the given table ().

Input: CPU load (L_1, L_2, \dots, L_n)
Output: predicted value
Process: <ol style="list-style-type: none"> 1. read N load values 2. find average load values using

3. $load = \frac{1}{N} \sum_{i=0}^N L_n$
4. create temp array
5. for $i=0$ to N
 - a. if $load \leq attribute_n$ then
temp[i]=1
 - b. else
temp[i]=0
 - c. end if
6. end for
7. find sum of each instances in row data in temp
8. update database with calculated distance
9. get current load parameters
10. compare and find distance
11. add classification label
12. calculate average of classified values
13. return value

In this proposed work required a GUI interface with user thus the proposed system is implemented using visual studio IDE which is contains a rich classes and libraries to implement desired functions and methods.

F. performance evaluation

In this section of the chapter given results are based on execution of algorithm which is implemented in background in given application. a number of experiment performed and most relevant records are provided here. The detailed description and performance of algorithm is given as:

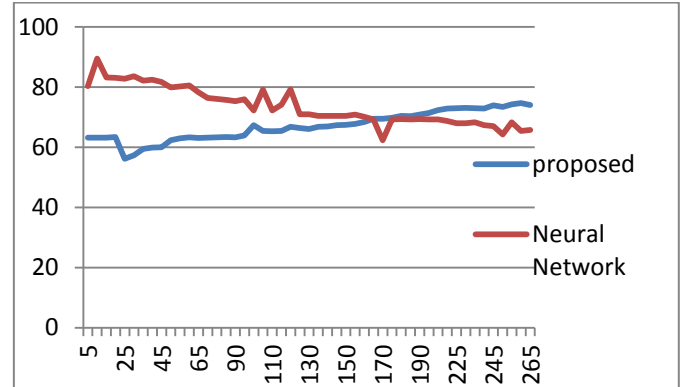


Fig3:Performance comparison between neural network and proposed system

1. Accuracy

The performance of the proposed algorithm and neural network is given in blow given figure () respectively. Accuracy of the system is estimated using the below given formula.

$$\% \text{ accuracy} = \frac{\text{total number of correctly classified}}{\text{total number of samples in input}} \times 100$$

The data increases in database the performance of the BPN algorithm not consistent found during experiments. But the proposed algorithm with less data the accuracy of the algorithm is adoptive and as data is increases the performance of the algorithm remains constant most of the time. Some of the parameters are remains to analysis and here an improvement can be adopted in future to improve efficiency more consistent and more efficient then neural network.

2. Error rate

Performance parameter error rate is indirectly proposanal to accuracy the performance of neural network and proposed algorithm is simulated using below given graph, whereas time increases the neural network performance is deceases and provided algorithm improve their performance of classification.

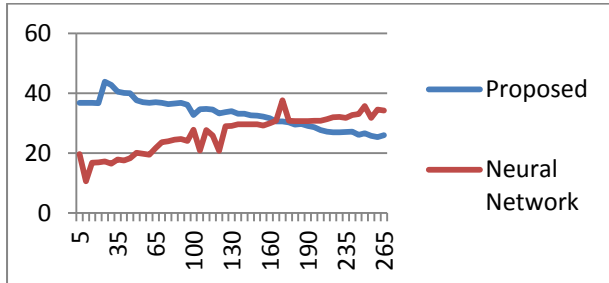


Fig4: systems error rate in %

The error rate is estimated as

$$\% \text{ error rate} = \frac{\text{total incorrectly classified data}}{\text{total number of data provided}} \times 100$$

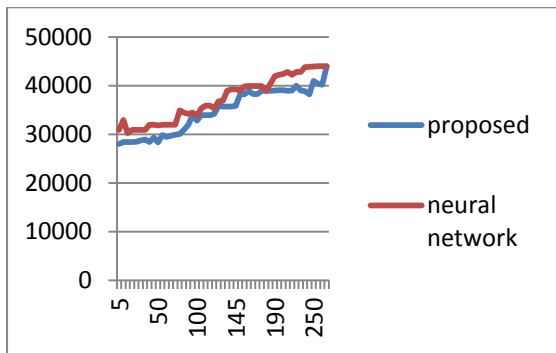


Fig5: Memory uses of the system

3. Memory uses

The memory uses of the system is given using the figure 5 where memory uses are most of the time remains constant and change in little bit data is found. In first look neural network and proposed algorithm consume similar memory size in terms of KB.

4. Build time and search time

Total elapse time among execution of the algorithm to build data model is known as build time. And total time required to predict new value from the system is defined as search time. Neural network consumes more cycles to train neural network from data base thus the build time of the system is quite high. In addition of the propose algorithm just analysis the data thus the build time of the system is less then neural network algorithm. Search time is also an important parameter and neural network predict values faster than the proposed algorithm to predict a value our proposed algorithm generate new values and update the data to the previous data table to classify data thus the time to predict new values consumes too less time.

5. PERFORMANCE DURING PREDICTION

the estimated load of the system is given using the below given figure which provides by the three different lines where red lines shows the real value of the system, blue line shows the proposed algorithms predicted values and green line shows the neural network values.

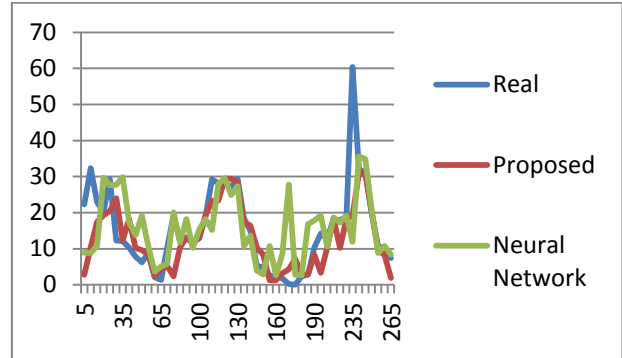


Fig6: load prediction

6. CONCLUSIONS

The presented section provides the complete work summary which includes all the concluded facts and suggestion to improve our task in future studies. Main aim of this proposed study to design and implement a novel and adoptive algorithm for classification and prediction of CPU load data. Load on CPU is uneven and random in nature that is directly depends upon the number of request per second. In addition of that it is also depends upon the number of processes running on host computer.

on the other hand the grid organization is depends upon two things network and CPU unit, thus there are two main kind of fault can occurred due to heavy traffic load network level or software conflicts level. To monitor these events two main techniques are suggested predictive and statistical analysis. We are studying predictive technique for monitor and manage the load on servers. Thus here introduces a new algorithm which is working with statistical calculation methods for predicted one step ahead values. The performance of the implemented technique and neural network is provided in above section and the performance summary is given using the below given table.

Parameters	Comments
Accuracy	Accuracy of the neural network classification is high but continuously deceases with number of instances on the other hand the performance of the proposed algorithm is improves during the time
Memory uses	Proposed algorithm and neural network consumes equivalent memory resource during execution
Search time	Search time of neural network is efficient than proposed method
Build time	Build time is less than the neural network algorithm

During the experiments we found that proposed algorithm is an adoptive technique to learn and able to predict load correctly but some performance improvements are required to much better performance of the implemented system.

In this proposed work to predict the CPU load system utilized the network and CPU load parameters to develop the predictive method, in future some more parameters are remain to explore. in addition of due to lake of correct parameters algorithm design is also effected in future requiredto estimate additional CPU load parameters are some logical corrections on algorithm is required by which performance of algorithm can improved more.

7. REFERENCES

- [1] Introduction to Grid Computing, Bart Jacob, Michael Brown, Kentaro Fukui, NiharTrivedi, ibm.com/redbooks
- [2] The Statistical Properties of Host Load (Extended Version), Peter A. Dinda, March1999 CMU-CS-98-175
- [3] Survey of Grid Resource Monitoring and Prediction Strategies, Liang Hu, Xiaochun Cheng, XilongChe, College of Computer Science and Technology, Jilin University.
- [4] FAULT TOLERANT SCHEDULING STRATEGY FOR COMPUTATIONAL GRID ENVIRONMENT, Malarvizhi Nanda gopalet. al. / International Journal of Engineering Science and Technology Vol. 2(9), 2010, 4361-4372
- [5] CPU Load Predictions on the Computational Grid, Yuanyuan Zhang, Wei Sun and Yasushi Inoguchi, Center for Information Science, Japan Advanced Institute of Science and Technology 1-1 Asahidai, Nomi, Ishikawa, 923-1292, Japan
- [6] Multi-model prediction for enhancing content locality in elastic server infrastructures, Juan M. Tirado, Daniel Higuero, Florin Isaila, Jesus Carretero, Computer Architecture and Technology Area, Universidad Carlos Madrid, Spain
- [7] Extended Forecast of CPU and Network Load on Computational Grid, SayakaAkioka, Yoichi Muraoka, 2004 IEEE International Symposium on Cluster Computing and the Grid
- [8] Host Load Prediction in a Google Compute Cloud with a Bayesian Model, Sheng Di, Derrick Kondo, WalfredoCirne, 978-1-4673-0806-9/12, 2012 IEEE
- [9] Agent Based Priority Heuristic for Job Scheduling on Computational Grids, Syed NasirMehmood Shah, M Nordin B Zakaria, Ahmad Kamil Bin Mahmood, AnindyaJyoti Pal, NazleeniHaron, International Conference on Computational Science2012
- [10] Jason R. Bowling, Priscilla Hope, Kathy J. Liszka, "Spam Image Identification Using an Artificial Neural Network", The University of Akron Akron, Ohio 44325-4003