Performance Evaluation of Predictive Replica Selection Using Neural Network Approaches

Shaik Naseera
Department of Computer Science & Engineering
Sri Venkateswara University
Tirupati, India
naseerakareem@gmail.com

K.V. Madhu Murthy
Department of Computer Science & Engineering
Sri Venkateswara University
Tirupati, India
kvmmurthy@yahoo.com

Abstract— **The ability to accurately predict a best replica from different sites holding replicas of a particular file is of great importance for applications that require access to replicated files for their execution. The best replica is the one that optimizes the desired performance criterion such as speed, cost, security or transfer time. As grid is dynamic in nature, the predicted best site for replica selection may not be the best site for replica selection with current network conditions. Neural network approaches address such dynamism and predict the best site more accurately for change in the network conditions. In this paper we compare and evaluate the prediction differences of various neural network approaches for replica selection problem.**

Keywords-Replica Selection, Candidate Site, Grid Computing, Neural Network Approaches, GridSim Toolkit-4.0.

I. INTRODUCTION

Data Replication is an important issue to be considered in a data grid environment. Applications like high energy physics, earth quake engineering produces large volumes of data sets which need to be stored and analyzed among users in the grid [14]. Data replication permit data sharing across many organizations in different geographically disperse locations [3]. Data Replication reduces the network bandwidth and access latency in grid environment. It also improves the reliability of the system by increasing data availability.

As different sites hold replicas of a particular file, there is a significant benefit realized by selecting a best replica among them. By selecting the best replica, the access latency time is minimized. The best replica will be the one that optimizes the desired performance criterion such as execution time, access cost and data transmission time [4].

As grid is dynamic in nature, the user requests, network latency, CPU load vary dynamically. Therefore the selected site to fetch replica may not be the best site for subsequent requests for change in the network conditions. Dynamism in such environments can be handled using neural network approaches to predict the behavior of the grid for subsequent requests.

In this paper we developed a predictive framework for different neural network algorithms like Back Propagation (BP), Batch Back Propagation (BBP), Back Propagation with momentum (BPM), Quick Propagation (QProp) and Resilient Propagation (RProp). Using these algorithms we predicted the transfer time of the sites that host replicas.

We evaluated the performance of these algorithms and demonstrated that Resilient Propagation algorithm is capable of predicting transfer time more accurately compared to other algorithms. This paper extends Rahman & Bakers [1] predictive technique for replica selection in grid environment. The replica selection is predicted using other neural network algorithms and evaluated their performances. The study of our replica selection algorithms is carried out using EU Data Grid Testbed1 [5] sites and their associated network geometry. We validated our model by using GridSim 4.1 simulation toolkit [6].

The reminder of the paper is organized as follows. We present the related work in section 2. the neural network architecture and various algorithms are discussed in section 3. The simulator and the simulation model are described in section 4. The performance evaluation and comparison of neural network approaches of replica selection problem is presented in section 5. Section 6 provides conclusions.

II. RELATED WORK

Grid is a dynamic environment in which the users requests, network latency, CPU load vary dynamically. The data in a data grid is replicated to share among the users spread across the network.. To execute a job, if all the files are currently available on a resource local storage, the job is sent to a resource's scheduler for execution. Otherwise the needed files are fetched from the nearest locations holding the replica. The ability to accurately predict the best replica is of great importance for applications which require replicated data to be accessed for their execution.

Observation from past application performance can also help in predicting present performance. Basu et a. [7] developed a time series model for internet traffic. They investigate the parametric time series models for aggregated data traffic. They developed an algorithm that predicts the u^{th} quantile of the distribution of data traffic given past history.

Wolski [14] designed network weather service (NWS) to predict network behavior of small file transfers. NWS is a distributed system that periodically monitors and dynamically forecasts the performance of network and computational resources over a given time interval.

978-1-4244-4711-4/09/$25.00 ©2009 IEEE

P.A.Dinda, D.R. Hallaaron [19] demonstrated that some applications, the relationship between the execution time of a CPU bound task and the measured load during the execution is almost perfectly linear. Therefore, load prediction is useful for guiding the scheduling strategies to achieve high application performance and efficient resource use [17,18].

Rashedul et. Al [1] made comparison of predictive replica selection using neural network with a multi-regression model. He demonstrated that the neural network technique is capable of predicting transfer time more accurately than the regression model.

In this paper we further extended the Rashedul et al [1] work for replica selection for other neural network approaches. We have done the comparison of various neural network approaches for replica selection problem.

III. Neural Network Approaches

The neural network tries to mimic the biological brain neural network into a mathematical model. It is a collection of simple processing units, mutually interconnected, with weights assigned to the connections. By modifying these weights according to some learning rule, the neural network can be trained to recognize any pattern given the training data. There are several types of neural network structures proposed in the literature. This network contains one input layer, one or more hidden layers and an output layer. The number of neurons in the input and output layers are governed by the number of inputs and outputs of the pattern to be recognized. However, the number of neurons in the hidden layer can be selected depending on the application. There are several types of training algorithms in the literature.

A. Learning Process

The primary significance of a neural network is the ability of the network to learn from its environment and to improve its performance through learning. The type of learning is determined by the manner in which parameter changes takes place. Learning algorithms in neural networks are generally categorized as supervised or unsupervised. In supervised learning, the target values or desired outputs are known and are given to neural network during training so that the neural network can adjust its weights to try to match its outputs to the target value or desired value. In unsupervised learning, the neural network is not provided with the correct results during training. Most applications fall within the domain of estimation problem such as statistical modeling, blind source separation and clustering.

In this paper, we are dealing with supervised learning algorithms like BP, BBP, BPM, QProp and RProp. The supervised learning rules are more useful for applications involving prediction, compression, digital signal processing and forecasting applications.

Each learning algorithm consists of two passes through different layers of the network: a forward pass and a backward pass. In forward pass an input vector is applied to the sensory nodes of the network and its effect propagates through the network layer by layer and the synaptic weights of the network are all fixed. In backward pass, the synaptic weights are adjusted in accordance with an error-correction rule [9].

A fully connected, multilayered perceptron with two hidden layers and an input layer is depicted in Fig. 1. Each unit in one layer is connected in the forward direction to every unit in the next layer. The input signal propagates through the network in forward direction, on a layer by layer basis and emerges as an error signal at the output end of the network. The error signal propagates backward layer by layer through the network. The hidden layer enables the network to learn complex tasks by extracting important features from the input vectors.

B. Back Propagation Algorithm :

Back propagation provides a computationally efficient method for the training of multiplayer perceptron [9]. The back propagation algorithm derives its name from the fact that the partial derivatives of the cost function with respect to free parameters (synaptic weights and biases) of the network are determined by back propagating the error signal through the network. The back propagation algorithm operates in one of the two modes: sequential or batch.

In the sequential mode the synaptic weights of all the neurons in the network are adjusted on a pattern-by-pattern basis. In batch mode the adjustments to all synaptic weights and biases are made on epoch-by-epoch basis with the result that a more accurate estimate of the gradient vector is used in the computation. A consequence of the back propagation algorithm is that there are situations where it can get stuck with a local minima and the algorithm is trapped and prevented to descend further. Some enhancements to the back propagation algorithm have been developed to get around this, one approach is back propagation with momentum.

Momentum adds a term in the weight adjustment that is proportional to the previous weight change. Once an adjustment is made it is remembered and serves to modify its subsequent weight adjustment. Momentum term avoids oscillation problems common with the regular back propagation algorithm when the error surface has a very narrow minimum area [10].

C. Quick Propagation Algorithm :

One method to speed up the learning is to use information about the curvature of the error surface. This requires the computation of the second order derivatives of the error function. Quick propagation algorithm assumes the error surface to be locally quadratic and attempts to jump in one step from the current position directly into the minimum of the parabola.

D. Resilient Propagation :

Resilient propagation algorithm is one of the most popular adaptive learning rates training algorithm [11]. It employs a sign-based scheme to eliminate harmful influences of derivatives magnitude on the weight updates, and is eminently suitable for applications where the gradient is numerically estimated or the error is noisy. The ideas behind the resilient propagation have motivated the development of several

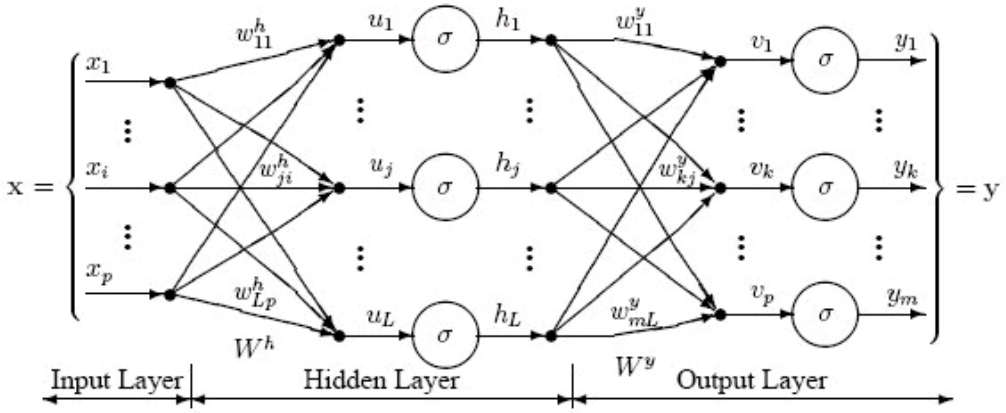

Figure 1. A fully connected multilayered perceptron for neural network

variants with the aim to improve the convergence behavior and effectiveness of the original method.

IV. SIMULATION

To evaluate our approach we use a simulation package called GridSim 4.1. GridSim toolkit is a data grid simulator. It provides the ability to define resources with heterogeneous storage components and the flexibility to implement various data management strategies like creation, deletion and replication of files in a grid environment [6].

A. *Architecture*

The simulation design consists of number of data resources each consisting of one or more processing elements with one or more storage elements. The processing elements provide computational capability and storage elements serves as data storage resource for submitted jobs. A replica manager handles and manages incoming requests about data sets in a resource for one or more storage elements. It also performs registration of files stored in the resource to a designated replica catalog (RC). The function of a RC is to store the metadata about files and to provide mapping between filename and its physical location.

A data-intensive job in a GridSim is termed as a DataGridlet. Each DataGridlet has a certain execution size in MI (Million Instructions) and require access to a set of files which may be located at different locations. After receiving a DataGridlet, the Replica Manager (RM) at each resource checks a list of required files for executing the job.

If all the files are currently available on a resource local storage, the DataGridlet is sent to a resource's scheduler for execution. Otherwise RM sends a request for obtaining the needed files from other resources. When all the requested files have been transferred and stored on the local storage, then the DataGridlet is executed by the scheduler.

B. *Grid Configuration*

The study of predictive technique for replica selection is carried out based on an EU Data Grid Testbed1 [5]. The resources and their associated network geometry are as shown in Fig 2. Initially all the master files are placed on the CERN (European Organization for Nuclear Research) storage. A master file is an original instance of the file.

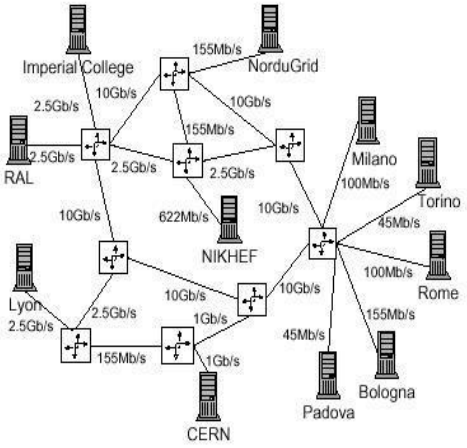

Figure 2. A simulated topology of EU Data Grid Testbed1

Table 1 summarizes all the resource relevant information. The resource's PE rating is modeled in the form of MIPS. We conducted this experiment for 100 files. The average file size is 1GB and the file size follows a power law distribution [15]. We have created 100 types of data intensive jobs. Each job requires 10 to 60 files to be executed. The required files for the DataGridlets are chosen with Zipf-like distribution [16].

The experiment is conducted with 200 DataGridlets with the approximate size of 84000KMI ± 30%. Each resource is provided with a user to submit their jobs approximately for every 1 minute. Each resource, peak load is chosen as specified in the Table 1.

TABLE I. RESOURCE SPECIFICATION

Resource ID	*Resource Name (location)*	*Storage (TB)*	*#Nodes*	*CPU Rating*	*Policy*	*Load*	*#User*
Res_0	RAL(UK)	2.75	41	49000	Space-Shared	0.82	24
Res_1	Imperial College(UK)	1.80	52	62000	Space-Shared	0.87	32
Res_2	NorduGrid(Norway)	1.00	17	20000	Space-Shared	0.69	8
Res_3	NIKHEF(Netherlands)	0.50	18	21000	Space-Shared	0.02	16
Res_4	Milano(Italy)	0.35	5	7000	Space-Shared	0.34	8
Res_5	Torino(Italy)	0.10	2	3000	Time-Shared	0.84	4
Res_6	Rome(Italy)	0.25	5	6000	Space-Shared	0.36	8
Res_7	Bologna(Italy)	5.00	67	80000	Space-Shared	0.34	24
Res_8	Padova(Italy)	0.05	1	1000	Time-Shared	0.24	4
Res_9	CERN(Switzerland)	2.50	59	70000	Space-Shared	0.42	48
Res_10	Lyon(france)	1.35	12	14000	Space-Shared	0.62	24

Some parameters are identical for all network links i.e., the Maximum Transmission Units (MTU) is 1500 bytes and the latency of links is 100msec. The RM of each resource follows least-frequently used replica strategy to delete a replica when the storage capacity is full. However, the master files at CERN can not be deleted or modified during the simulation.

V. SIMULATION FRAMEWORK

As grid is dynamic in nature, the predictive best site for replica selection may not be the best site for replica selection for change in the network conditions. It is shown in [1] that, neural network technique is capable of predicting transfer time more accurately than the regression model. We further extend the concept to compare the prediction differences of different neural network algorithms.

To evaluate the performance of different learning algorithms for replica selection problem, we use a neural network consisting of one input layer, one output layer and multiple hidden layers. The input layer consists of 2 neurons, an output layer consists of one neuron and varying number of neurons in the hidden layers. We set up the EU data grid testbed1 [5] using GridSim toolkit. During the simulation, when a file transfer has been made between two sites, the file size, the available network bandwidth and transfer time are saved which is later used for training and testing the neural network.

The neural network is trained with a training set consist of three values i.e., file size, available bandwidth experienced between sites to transfer the file and time elapsed to transfer the file. We start training the neural network with 20 datasets. Each set is presented to the neurons of the input layer and the whole network is trained with the learning algorithms separately and the results are compared. The output of the neural network predict the transfer time between the sites. After training the neural network with first 20 datasets from the training pattern, the prediction is made for the datasets in the testing pattern.

To compare the prediction differences of the algorithms, the neural network is trained for more than 10000 epochs with a learning rate of 0.2. Learning rate in neural network determines how much we change the weights in each step. The smaller we make the learning rate parameter, the smaller the changes to the synaptic weights in the network from one iteration to the next and smoother will be the trajectory in weight space and algorithm converges. On the other hand too large learning rate parameter makes the neural network unstable, bouncing around the error surface, so the algorithm diverges.

As various components in the grid environment are dynamic, the file transmission time between two sites varies according to change in the network conditions. To address the dynamic nature of the grid, we train the neural network with a smaller momentum term value of 0.5.

A. *Simulation Results :*

The performance analysis of neural network algorithms like BP, BBP, BPM, QProp and RProp for replica selection problem is done for a neural network consists of one input layer, one output layer and varying number of hidden layers. The Fig. 3 depicts the error graph of BP, BBP, BPM, QProp and RProp algorithm for different multi-layered neural network architectures.

Fig 3 shows that the batch mode back propagation takes a much longer time to converge as it takes the total training error over all the patterns into account whereas standard back propagation makes estimation of the error based on the individual error training pattern. It is also observed from Fig 3 that BPM converges faster than BP, BBP as the term momentum is helpful in speeding up the convergence and avoid local minima.

The QProp algorithm takes lesser time to converge, as it uses the second order derivatives of the error function to jump in one step from the current position directly into the minimum of the error performance and hence it is faster than the backpropagation algorithms. As depicted in Fig 3. RProp is demonstrating better performance compared to BP, BBP, BPM and QProp algorithms. It takes less number of cycles to converge and optimize the error rate.

The number of training cycles and their optimal error rate for learning algorithms for neural network with varying number of hidden layers is tabulated as shown in Table 2.

The experiment is conducted and performance is analyzed for replica selection problem for different hidden layers in the neural network. To minimize the error rate, the neural network is trained for different cycles for different learning algorithms. The optimal error rate is obtained for the different algorithms at different training cycles.

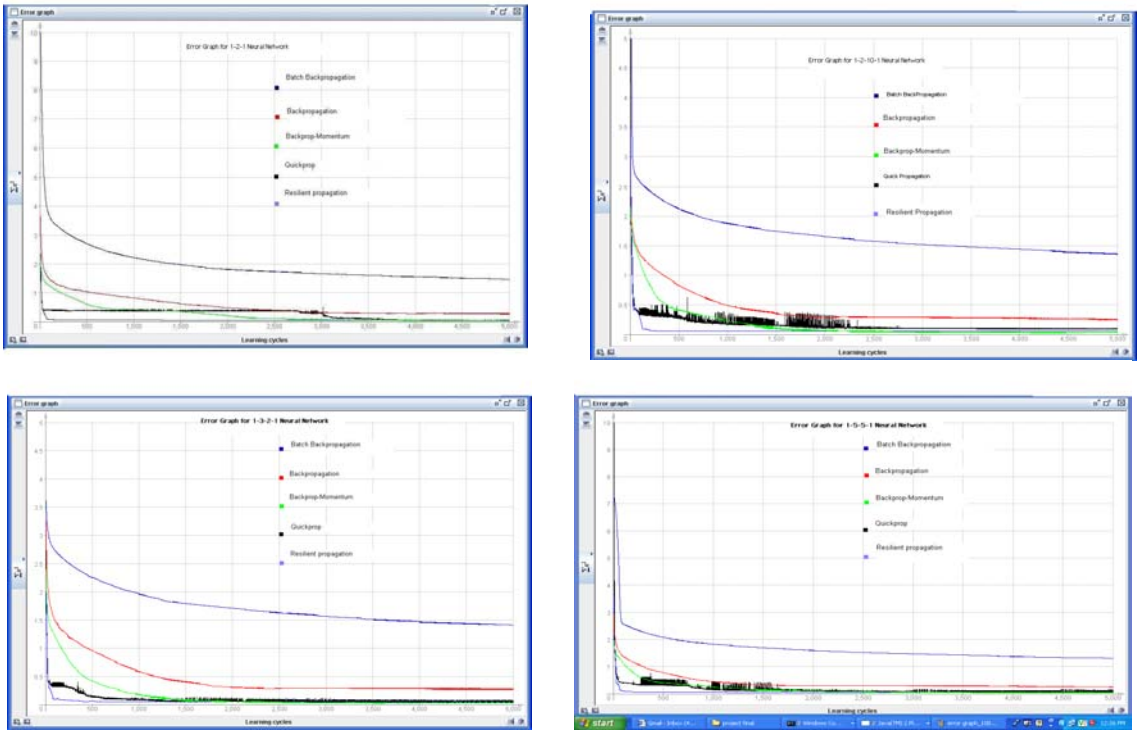

Figure 3. Error Graph for Multi-Layered Neural Network Architectures

TABLE II. TRAINING CYCLES REQUIRED FOR VARIOUS NEURAL NETWORK ALGORITHMS TO GET OPTIMAL ERROR RATE

TRAINING CYCLES FOR 1-2-1 NEURAL NETWORK

Algorithm	*Cycles*	*Error Rate*
BP	86500	0.0
BBP	100000	0.407
BPM	13650	0.0
QProp	34000	0.038
RProp	17610	0.05

TRAINING CYCLES FOR 1-2-10-1 NEURAL NETWORK

Algorithm	*Cycles*	*Error Rate*
BP	75000	0.0
BBP	100000	0.278
BPM	24000	0.0
QProp	28000	0.045
RProp	41550	0.048

TRAINING CYCLES FOR 1-5-5-1 NEURAL NETWORK

Algorithm	*Cycles*	*Error Rate*
BP	75600	0.0
BBP	100000	0.266
BPM	20900	0.0
QProp	28000	0.078
RProp	33650	0.048

TRAINING CYCLES FOR 1-3-2-1 NEURAL NETWORK

Algorithm	*Cycles*	*Error Rate*
BP	51800	0.0
BBP	100000	0.308
BPM	11000	0.0
QProp	28000	0.038
RProp	29400	0.0488

TABLE III. PREDICTION TIME DIFFERENCES OF ALGORITHMS WITH TRAINING RATE OF 10000 EPOCHS FOR 1-2-1 NEURAL NETWORK

Res_ID	*GridSim Transfer Time*	*BP(0.093)**	*BPM(0.025)**	*BBP(1.26)**	*QProp(0.03)**	*RProp(0.052)**
Res_0	188	81	67	64	92	17
Res_2	106	73	1	16	58	8
Res_4	426	53	32	24	3	13
Res_6	455	63	11	7	23	3
Res_7	64	54	36	46	42	6
Res_8	660	37	59	147	60	21
Res_9	264	47	96	83	96	39

*error rate of the algorithms at 10000 epochs

TABLE IV. PREDICTION TIME DIFFERENCES OF ALGORITHMS WITH TRAINING RATE OF 10000 EPOCHS FOR 1-5-5-1 NEURAL NETWORK

Res_ID	*GridSim Transfer Time*	*BP(0.0616)**	*BPM(0.0288)**	*BBP(1.1109)**	*QProp(0.0416)**	*RProp(0.5055)**
Res_0	188	99	88	21	106	15
Res_2	106	64	39	58	82	6
Res_4	426	31	7	44	10	15
Res_6	455	46	12	28	27	5
Res_7	64	23	16	58	58	3
Res_8	660	45	78	108	47	19
Res_9	264	79	97	64	95	37

*error rate of the algorithms at 10000 epochs

The experimental results shows that, the number of epochs needed for training the neural network is more for BBP and is less for QProp. The prediction difference for file transmission time between the sites by BP, BBP, BPM, QProp and RProp is shown in the 3. The tabulated data shows that the RProp is predicting the more accurate transfer time compared to other algorithms. Though the RProp takes more training cycles compared to QProp, the prediction difference from the actual value is very small.

The experimental results shows that, the number of epochs needed for training the neural network is more for BBP and is less for QProp. The prediction differences of algorithms for file transmission time between the sites by BP, BBP, BPM, QProp and RProp is shown in the Table 3 and Table 4. The tabulated data shows that the RProp is predicting the transfer time between the sites with small variation from the GridSim transfer time compared to other algorithms.

VI. CONCLUSIONS

The performance evaluation of different neural network approaches for replica selection problem is done. The neural network approaches predict the site to fetch the replica with more accuracy of transfer time prediction. We analyzed the performance measure in terms of accuracy among neural network approaches for predicted transfer time between sites that hold replica currently.

The training time and error rate differences among the neural network approaches are presented. We evaluated the performance of these algorithms and demonstrated that Resilient Propagation algorithm is capable of predicting transfer time more accurately compared to other algorithms.

REFERENCES

[1] Rashedur M. Rahman, Ken Baker, Reda Alhajj, A Predictive Technique for Replica Selection in Grid Environment, Seventh IEEE International Symposium on Cluster Computing and the Grid (CCGrid 07), 2007.

[2] Rahman, M. Rahman , Ken Baker and Reda Alhajj., Replica Placement Design with Static Optimality and Dynamic Maintainability, Proceedings of the IEEE/ACM International Conference on Cluster Computing and Grid (CCGRID 06), Singapore, May 2006.

[3] Kavitha R and Foster I ., Design and Evaluation of Replication Strategies for a High Performance Data Grid, Proceedings of computing and high energy and nuclear physics, 2001.

[4] A. Chervenak, I. Foster, C.Kesselman, C. Salisbury, and S.Tuecke. The data grid : Towards an architecture for the distributed management and analysis of large scientific datasets. Journal of Network and Computer Applications, (23): 187-200, October 2000.

[5] The European Data Grid Project homepage : http://eu-datagrid.web.cern.ch/eu-datagrid,2005.

[6] Anthony Sulistio, Uros Cibej, Borut Robic and Rajkumar Buyya., A Toolkit for Modelling and Simulation of Data Grids with Integration of Data Storage, Replication and Analysis, Journal of Elsevier Science, 2006.

[7] Basu S.A Mukherjee and S. Kilvansky, Time Series Models for Internet traffic, 1996, Georgia Institute of Technology.

[8] Simon Haykin, Neural Networks, A comprehenssive foundation, second edition, 1999.

[9] Marvia Minsky and S Pappert, Neural Networks for pattern recognition, MIT Press, 1969.

[10] Fausett L., Fundamentals of Neural Networks : Architecture, Algorithms and Applications. Prentice Hall, Upper Saddle River, NJ, 1994.

[11] M. Riedmiller and H.Braun, A direct adaptive method for faster backpropagation learning: The RProp algorithm, International coneference on neural networks, San Fransisco, 581-591, 1993.

[12] M.Pfister and R Rojas, QProp – A hybrid learning algorithm which adaptively includes second order information, Proc. 4th Dortmund Fuzzy Days, 55-62, 1994.

[13] Prechelt L, P ROBEN1 – A set of benchmarks and benchmarking rules for neural network training algorithms, Technical Report 21/94. Fakultat fur Informatik, University of Karlsruhe, 1994.

[14] R. Wolski. Dynamically forecasting network performance using the network weather service, journal of cluster computing Vol. 1, pp 119-132, 1998.

[15] W. Gong, Y. Liu, V. Misra and D. Towsley. On the tails of web file size distributions. In Proceedings of 39th Allerton Conference on Communication, Control and Computing, November 2001.

[16] L. Breslau, P. Cao, L. Fan, G. Phillips and S. Shenkar. Web catching and zipf-like distributions: Evidence and implications. In INFOCOM (1), pages 126-134, 1999.

[17] P.A. Dinda, A predictive based real-time scheduling advisor, Proceedings of 16th international parallel and distributed processing symposium (IPDPS 2002) pages 35-42, 2002.

[18] L. Yang, J.M. Schops and I.Foster, Conservative scheduling : Using predictive variance to improve decisions in dynamic environment, Supercomputing '03, pages 1-16, 2003.

[19] P.A. Dinda, D. R. O'Hallaron, Host load prediction using linear models, cluster computing 3(4) pages 265-280, 2000.