International Journal Of Computer Science And Applications Vol. 1, No. 2, August 2008
ISSN 0974-1003

Host Load Prediction Using Adaptive Models

Pallav J Gandhi
Maharashtra Academy of Engineering, Alandi), Pune-412105, (MS) India. +91-020-27185857
gandhi.pallav@gmail.com

Sachin M Munot
Maharashtra Academy of Engineering, Alandi), Pune-412105, (MS) India. +91-020-27185857
suchit_patel@yahoo.com

Suchit A Patel
Maharashtra Academy of Engineering, Alandi), Pune-412105, (MS) India. +91-020-27185857
sachin.munot@gmail.com

Noor A Memon
Maharashtra Academy of Engineering, Alandi), Pune-412105, (MS) India. +91-020-27185857
noor.876@gmail.com

ABSTRACT
Desktop grids are more popular platform for high throughput applications, but due to their inherent resource volatility it is difficult to exploit them for the applications that require rapid turnaround. Efficient resource management for short lived applications is required to implement desktop grids at institute or enterprise level. For efficient resource management, host load prediction is a prime concept in deciding the time taken by a particular resource or host to run the current task. On the basis of running time of task, load of the server can be distributed to the available resources efficiently.

Already proposed prediction schemes which use Linear Models like AR, MA, ARMA can predict the host load consistently to some extent. AR(18) model provides a confidence interval of less than 200 ms for up to 25 seconds in to the future (when used to predict the running time of a one second task) which is equivalent to 20% of the host load. Performance of this schemes decreases for heavy or long term host loads as they introduces very long confidence intervals. In desktop grid networks length of the confidence interval should be minimum possible such that 95% of the tasks would have running time in their predicted intervals. Hence we propose Adaptive Schemes for the prediction of host loads which introduces confidence interval less than 0.1% even for 30 seconds into the future. This paper discusses the importance and simulation results of prediction using LMS Adaptive Estimator.

Categories and Subject Descriptors
[Parallel and Distributed Computing]: Grid Networks, Desktop Grid, Distributed Computing

General Terms
Algorithms, Measurement, Performance, Design.

Keywords
Desktop Grid, Adaptive Prediction, Host Load Prediction.

1 INTRODUCTION
Desktop grids use the idle cycles of mostly desktop PC's to support large-scale computation and data storage. Today, these types of computing platforms are the largest distributed computing systems in the world. The most popular project, SETI@home, uses over 20 TeraFlop/sec provided by hundreds of thousands of desktops. Indeed, it is a well-studied fact that machines primarily devoted to regular human-dependent interactions, like e-office applications (work processing, spreadsheets, etc.) and e-communications such as instant messaging, e-mail, and Internet browsing barely use their resources. For instance, Heap [5] reports nearly 95% CPU idleness amongst Unix machines, with an even higher value of 97% measured in Windows machines assigned to academic classrooms[14]. Furthermore, in their comprehensive study of more than 200000 SETI@home [16] hosts, Anderson and Fedak [6] report that an average of 89.9% of CPU was volunteered to public computing through the BOINC platform [7], meaning that roughly 90% of CPU would have been wasted if it was not exploited by BOINC projects.

Despite the popularity and success of many desktop grid projects, the volatility of the hosts within desktop grids has been poorly understood. Due to lack of resource management techniques, application of desktop grid is limited to high throughput applications that consist of very large number of tasks. But in an enterprise or institute most of the time applications consist of small or moderate number of tasks. In this case desktop grids are underutilized. So it's a big issue of research that how to use grid networks effectively for *short lived application.* From results of [9], it is clear that for short live application there must be some resource selection scheme which can improve the performance. There are various algorithms like FCFS-AT, FCFS-TR, FCFS-PRDCT-DMD as explained in [9]. But to provide consistent high performance when running on typical shared, unreserved distributed computing environments, adaptive applications must exploit the degrees of freedom, by carefully choosing how and where to run the tasks. To make such decisions, applications require predictions of the performance of each of the alternatives. If the application could predict the running time of the task on each of the available hosts, it could trivially choose an appropriate host to run the task. According to Dinda [13], running time of the task depends upon the host load. Ultimately host load prediction can provide the proper information to select the appropriate host for efficient grid network. *This paper provides a novel idea of predicting host*

loads using Adaptive Model which is having far better performance than linear models used by Dinda [13].

The rest of this paper is organized as follows. Section II describes the concept of host load and running time prediction. Section III describes related research work. Section IV presents the idea to use adaptive algorithm and simulation results. Section V concludes the work.

2 AN IMPORTANT CONCEPT: HOST LOAD PREDICTION

2.1 Prediction of Running Time of Task

If the application could predict the running time of the task on each of the available hosts, it could trivially choose an appropriate host to run the task. But because of dynamic nature of hosts it is not possible to predict the exact running time of the task. There will be some error in prediction. The mean squared error in prediction is directly comparable to raw variance and with a normality assumption can be easily translated into confidence interval [13]. Nature of confidence interval should be such that 95% of the task should be finished within the confidence interval. So, *Better prediction results in smaller confidence interval which makes it easier for the application to choose between hosts.*

2.2 Relation between running time of task and host load

The running time of a task on a particular host is strongly related to the load on the particular host at given instance of time. If we can predict a load on the host we can predict the running time of task accordingly. Better load prediction results into better prediction of running time of a task. Better load prediction can also result into drastically smaller confidence intervals. This is why Host Load Prediction is an important concept for resource selection in desktop grid networks.

3 HOST LOAD PREICTION USING LINEAR MODELS

Host load prediction can be done using different methods. Quality of prediction decides how tight we can bound the running time of the tasks. Usually prediction quality is measured in terms of *mean squared error*, which is the average of square of difference between predicted value and original value. Mean squared error can further be translated in to confidence interval. If we go for k step ahead prediction, for different values of k, mean square error will be different. So predictability and confidence interval for different values of k will be different. One simplest method of prediction is long term mean method. In this method mean square error will be simply variance of a signal. Performance of this method is very poor as per the results shown in [13]. Another method used by researchers is prediction using Linear Models. According to Dinda and Hallaron [13] host load prediction can be done consistently using linear models like AR, MA, ARMA. They began by choosing to measure the host load by the Digital Unix five second load average. This measure can be easily acquired by user level program and is closely related to short live applications. They evaluated the performance of linear models for predicting Digital Unix five second host load average from 1 to 30 seconds into the future.

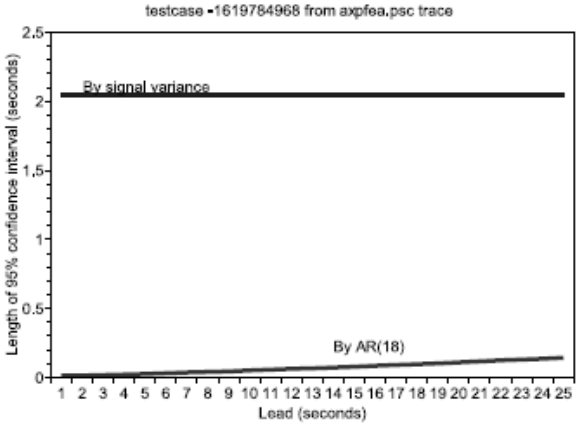

Figure 1. Benefits of prediction using linear models over raw variance method. [12]

Figure 1 plots the length of confidence interval for the running time of a one second task as a function of how far load predictions are made. Two different methods are used for prediction purpose, one is long term mean of the signal and another is AR(18) method. Notice that up to 25 seconds into the future AR(18) model provides a confidence interval of less than 200 ms where as long term mean of the signal method provides confidence interval of more than 2 seconds. Thus performance of linear model AR(18) is far better than the performance of long term mean method. Other linear models like MA, ARMA, ARIMA are evaluated for different load traces. Performance of MA model is poor as compare to AR model. ARMA,ARIMA improves the performance at some what extent but at the cost of computational complexity. So, AR model of order higher than 16 is suggested by Dinda and Hallaron.

4 PREDICTION USING ADAPTIVE MODELS

4.1 Problems with AR Model

As figure 1 shows AR(18) model introduces confidence interval of 200ms up to 25 seconds in to the future for the task having running time of one second. Length of the confidence interval is almost 20% of the original running time of task in this case. For long term or heavy load applications it will introduce very large confidence interval which is not desirable in the desktop grid networks. Again for AR models *mean square error* increases as value of k increases. This introduces irregularities in Predictability. So, we need such a method which introduces low confidence interval. In this paper we introduce Adaptive Model for prediction of host load. Error performance of Adaptive model is better than AR model. If we convert error in prediction directly into confidence interval than length of confidence interval can be reduced significantly using Adaptive Models.

4.2 Defining confidence interval in terms of Estimated Error in Prediction

Here we define confidence intervals in terms of maximum error in prediction. Length of *Confidence Intervals* will depend on the error in prediction. Let us say

Confidence Interval C= [T*i* , T*u*] (4.1)

Length of *Confidence Interval* T = T*u*-T*i*. (4.2)

We can define the relationship between *Maximum Error In Prediction (ξε)* and *Confidence Interval* as follows:

T*i* = τ – ξε. (4.3)

T*u*= τ + ξε. (4.4)

Length of *Confidence Interval*= T*u*-T*i*

= (τ + ξε) – (τ – ξε)

= 2ξε

$$\boxed{T = 2\xi\varepsilon} \quad (4.5)$$

From given formula, if we have maximum error in prediction we can calculate the *length of confidence interval*. To know the prediction error in advance we can again use the adaptive model. Using LMS Estimator we can estimate the error in prediction in advance. Let us say *estimated error in prediction is δε.*

So, if we replace maximum error in prediction ξε by estimated error in prediction $\delta\varepsilon$, equation (4.5) becomes

$$\boxed{T = 2\ \delta\varepsilon} \quad (4.6)$$

4.3 Simulation Results in MATLAB

As it is described in section 4.2, we can convert length of the confidence interval into estimated error. If we have a good predictor than length of the confidence interval can be reduced by a huge amount.

For example if maximum error in prediction ξε is 1% than estimated error in prediction $\delta\varepsilon$ will be 1.01% because to estimate the error in prediction we will use the same estimator. From equation (4.6) length of the confidence interval will be

$$\boxed{T = 2*[(\tau * \%\delta\varepsilon)/100]} \quad (4.7)$$

i.e. $T = 2*[(\tau * 1.01)/100]$

Here we have designed a LMS Adaptive Estimator of order 8 which predicts up to 30 seconds in future. To simulate the LMS Estimator we have used a database of almost 1500 tasks (n = 1500) having average running of one second.

Figure 2 shows the database of almost 1500 tasks having average running time of one second.

Figure 3 shows, the error performance of LMS Estimator which can predict up to 30 seconds in future. Maximum error in prediction is 1.6 ms, while predicting for the running time of one second. This error is very low as compare to the AR(18) model. Maximum error is almost 1.6 % , so resulting length of the confidence interval is only 3.2 ms which is very low as compare to 200ms of AR(18) model.

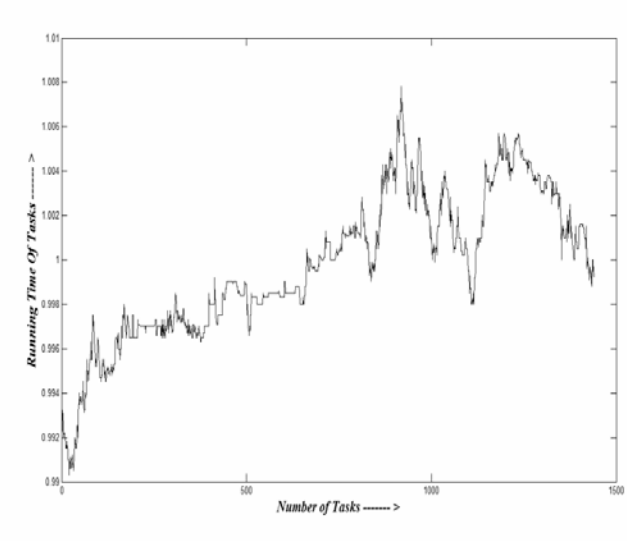

Figure 2. Running Time of almost 1500 tasks

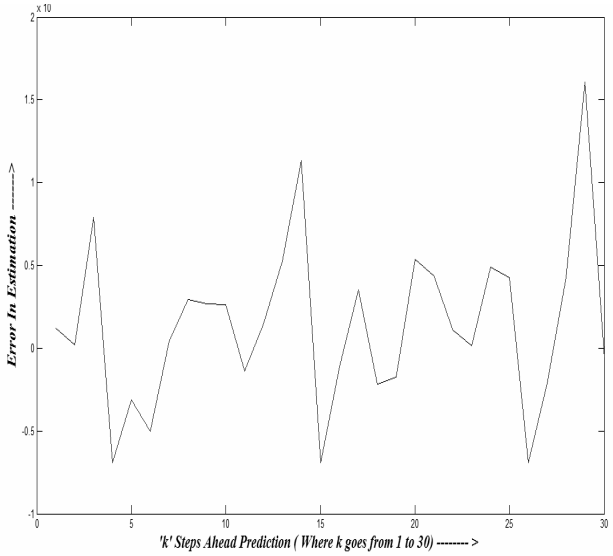

Figure 3. show the error in prediction when n = 35 for given database.

5 CONCLUSION

Thus in this paper we have designed a LMS Estimator which can predict up to next 30 seconds in to the future. LMS Estimator can predict with better error performance as compare to linear model AR(18) which is used previously. AR(18) provides the confidence interval of 200 ms for up to 30 second into the future while predicting the running time of task of one second which is almost 20%. Here newly approached LMS Estimator provides the confidence interval of 0.3% for the same which is the most desirable feature for the host load prediction for desktop grids. Length of the interval can further be reduced for long running time of a task. If the average running time of tasks is five to ten

seconds with the same spread than length of the interval can be reduced to less than 0.1%.

6 ACKNOWLEDGEMENT

This paper is based upon the research work supported by Signal Processing and Networking Society of MAE-IEEE Student Branch.

7 REFERENCES

[1] A. Andrzejak, P. Domingues, and L. Silva, "Classifier-based Capacity Prediction for Desktop Grids," presented at Integrated research in Grid Computing - CoreGRID workshop, Pisa,Italy.

[2] Berman, F. and R. Wolski: 1996, 'Scheduling From the Perspective of the Application'.In: Proceedings of the Fifth IEEE Symposium on High Performance Distributed Computing HPDC96. pp. 100–111.

[3] Bernard Widrow and Samuel D. Stearns, Adaptive Signal Processing, Pearson Education, 2004.

[4] BYTE, "BYTEmark project page (http://www.byte.com/bmark/bmark.htm)," Byte, 1996

[5] D. G. Heap, "Taurus - A Taxonomy of Actual Utilization of Real UNIX and Windows Servers," IBM White Paper GM12-0191, 2003.

[6] D. Anderson and G. Fedak, "The Computational and Storage Potential of Volunteer Computing," 2005.

[7] D. Anderson, "BOINC: A System for Public-Resource Computing and Storage," presented at 5th IEEE/ACM International Workshop on Grid Computing, Pittsburgh, USA., 2004.

[8] "Distributed Computing Info (http://distributedcomputing.info/)," 2006.

[9] D. Kondo, A. Chien, and H. Casanova, "Resource management for rapid application turnaround on entreprise desktop grids," presented at 2004 ACM/IEEE conference on Supercomputing, 2004.

[10] Dinda, P., B. Lowekamp, L. Kallivokas, and D. O'Hallaron: 1999, 'The Case for Prediction-Based Best-Effort Real-Time Systems'. In: Proc. of the 7th International Workshop on Parallel and Distributed Real-Time Systems (WPDRTS 1999), Vol. 1586 of Lecture Notes in Computer Science. San Juan, PR: Springer-Verlag, pp. 309–318. Extended version as CMU Technical Report CMU-CS-TR-98-174.

[11] E.C.Ifeachor and B.W.Jervis, Digital Signal Processing-A Practical Approach, Pearson Education, 2005.

[12] Monson.H.Hayes, Statistical Digital Signal Processing and Modeling, John Wiley Publication, 2001.

[13] P.A.Dinda and D.R. O'Hallaron:2000,'Host Load Prediction Using Linear Models', Cluster Computing3(4). An earlier appeared in HPDC'99.

[14] P. Domingues, P. Marques, and L. Silva, "Resource Usage of Windows Computer Laboratories," presented at International Conference Parallel Processing (ICPP 2005)/Workshop PEN-PCGCS, Oslo,Norway, 2005.

[15] Simon Haykin, Adaptive Filter Theory, Pearson Education, 4th edition, 2002.

[16] SETI, "SETI@Home Project (http://setiathome.berkeley.edu/)," 2005.