Extended Forecast of CPU and Network Load on Computational Grid

Sayaka Akioka Yoichi Muraoka
Science and Engineering School of Waseda University,
3-4-1 Okubo, Shinjuku, Tokyo, Japan, 169-8555
{akioka, muraoka}@muraoka.info.waseda.ac.jp

Abstract

To achieve effective load balancing and a robust Grid environment, extended load forecast for computational resources is increasingly required. Thus, this paper proposes 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 our algorithm is in using seasonal load variation for both load variance and one-step-ahead prediction. We apply seasonal fluctuation in CPU load to network load variation especially for network load variance prediction. Furthermore, the Markov model-based meta-predictor is used for one-step-ahead prediction, which is sensitive to late trends. The results of the experiments demonstrate that our algorithm gives a good curve for expected 8-day-long load variance, and makes accurate one-step-ahead predictions. The 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. Moreover, the least mean error rate for wider range forecasts is 5.5% for network load variation, and 3.6% for CPU load variation.

1. Introduction

A cluster of PCs or workstations scattered geographically worldwide and connected to each other by a network, which is called computational Grid [1], is becoming significant. To make Grid environment greatly useful, a scheduling system [2]-[7] is essential because computational Grid is often a heterogeneous environment. Generally, a scheduling system periodically gathers the load information of computational resources, such as CPU, network, memory, and disks, and then decides how to allocate applications based on the gathered information. In the Grid environment, however, it is difficult to keep this information up-to-date for the amplitude of the Grid environment. This implies that predicting CPU and network load, which waves frequently, is useful in obtaining good scheduling [8]-[10]. Moreover, this prediction is also useful in predicting turnaround time of application, which is one of the most important information items for the scheduling system [11]-[13], because the CPU and network are most essential elements of computational resources. At the same time, predicting load is useful in stably managing the Grid environment itself.

There are some projects on load prediction in the Grid environment [14]-[16]. However, the prediction range is around 10 seconds, and this is very short compared with the executing times of applications supposed to run on the computational Grid. We need not only short-term, precise prediction, but also mid-term and long-term prediction, for example, from hours and days to a longer range.

For this reason, this paper proposes an algorithm that predicts CPU load and network load. Our approach is to separate seasonal variation in CPU load for extended forecast of CPU load, and to apply seasonal variation for the extended forecast of network load. In addition, we use the Markov model-based meta-predictor [17] for precise one-step-ahead prediction. In the experiments, our method accurately predicts waves of CPU load and network load over a week, and the mean error rate for one-step-ahead prediction was 6.2% for CPU load and 9.4% for network load.

2. Related work

First, the Network Weather Service (NWS) [14] is one of the most famous projects on load forecast in the Grid environment. The meta-predictor is characteristic of NWS. NWS periodically monitors load, runs several prediction algorithms, and prepares several predictions according to each algorithm. At every prediction, the meta-predictor selects one prediction that is expected to be the most precise based on past prediction results of several prediction algorithms. NWS gives usage prediction of CPU, network, and memory, but its prediction range is only about 10 seconds, that is, one-step-ahead prediction.

Dinda et al. [16] use a linear model to predict mean CPU load for 5 minutes. However, the range of prediction is also in seconds.

Yang et al. [18] use periodically monitored CPU load to calculate 1) one-step-ahead prediction, 2) mean average CPU load for expected application execution time on a node, and 3) standard deviation over the approximated application execution time, and they then propose a scheduling method based on these predictions. The prediction of 2) and 3) is indispensable information to minimize the turnaround time of applications. The application execution time varies from 1 minute to 10 minutes. However, to thin out a large amount of available nodes to allocate applications, we need to know the qualitative load variance of each node.

Dinda et al. and Yang et al. predict only CPU load. However, in the Grid environment, a node often needs to cooperate with other nodes over the network and has to handle streaming media. This means that only CPU load prediction is inadequate for scheduling and we also need to at least know network load variance. Moreover, the range of all predictors is just in seconds or minutes. Assuming Grid applications are run with network access, this range is too short.

3. CPU load prediction

3.1. Extended forecast of CPU load

It is highly possible to find periodicity in waves of CPU load in the long term. Generally, time series data can be separated into 4 variations: 1) variation in the range of years, that does not circulate (trend variation), 2) variation in the range of years that does circulate (cyclical variation), 3) cyclic variation in the shorter range than cyclical variation (seasonal variation), and 4) variation without any trends (irregular variation). In this paper, the range is short enough to take no account of trend variation or cyclical variation. Therefore, along this separation, CPU load can be expressed by following expression. Here, $Y_{cpu}(t)$ measures CPU load at time t, M_{cpu} is mean value of measured CPU load, $\{S_i(t),\ i=1,2,\cdots,N\}$ are seasonal variations, and $I_{cpu}(t)$ is irregular variation.

$$Y_{cpu}(t) = M_{cpu} + S_1(t) + \cdots + S_N(t) + I_{cpu}(t)$$

In this paper, we assume $I_{cpu}(t)=0$ for extended forecast because the purpose of extended forecast is to predict the load variance, not to indicate a precise value. Therefore, all we have to do for extended forecast is to calculate M_{cpu} and separate $S_i(t)$. With the definition of $y_{cpu}(t) = Y_{cpu}(t) - M_{cpu}$, the moving average method separates $S_i(t)$. When d_i is the seasonal cycle of $S_i(t)$, $m_i(t)$, which is the result of separation of $S_i(t)$ from $y_{cpu}(t)$, it is calculated by following formula.

i) case $d_i = 2q$

$$m_i(t) = \frac{1}{d_i}\left(\frac{y(t-q)}{2} + \sum_{k=t-q+1}^{t+q+1} y(k) + \frac{y(t+q)}{2}\right)$$

ii) case $d_i = 2q+1$

$$m_i(t) = \frac{1}{2q+1}\sum_{k=-q}^{q} y(t-k)$$

If n is the maximum value of t, the series of seasonal variation in $S_i(t)$, $s_i(r)$ can be estimated by following formula, where $\{w_{i,r},\ r=1,\cdots d_i\}$ is the mean value of deviation of $y_{cpu}(t)$ and $m_i(t)$, and the deviation can be calculated by $y_{cpu}(r+j*d_i) - m_i(r+j*d_i)$, which satisfies $q < r + j*d_i \le n-q$.

i) case $r = 1,\cdots,d_i$

$$s_i(r) = w_{i,r} - \frac{1}{d_i}\sum_{k=1}^{d_i} w_{i,k}$$

ii) case $r > d_i$

$$s_i(r) = s_i(r-d_i)$$

Then, $FS_i(t)$ is an approximation of $s_i(r)$ using the least squares method, and extended forecast of CPU load $EP_{cpu}(t)$ can be expressed by following formula.

$$EP_{cpu}(t) = M_{cpu} + FS_1(t) + \cdots + FS_N(t)$$

In the evaluation in this paper, we assumed seasonal variation cycles to be 24 hours and 1 week because CPU load variation is strongly related the life style of the user.

3.2. One-step-ahead CPU load prediction

While extended forecast is mainly used to understand load variation tendencies or average load, one-step-ahead prediction is required to be accurate and sensitive to recent changes in measurements. Therefore, for one-step-ahead prediction, we use the Markov model-based meta-predictor to estimate irregular variation $I_{cpu}(t)$ in addition to the seasonal variation used for extended forecast. That is, $I_{cpu}(t)$ is expected to express recent fine changes. As a result, one-step-ahead prediction $SP_{cpu}(t)$ is expressed by following formula, where $EI_{cpu}(t)$ is the estimated irregular variation according to the Markov model-based meta-predictor.

$$SP_{cpu}(t) = EP_{cpu}(t) + EI_{cpu}(t)$$

The meta-predictor is a selector of the predicting algorithm at every prediction time. Each predicting algorithm calculates the prediction value itself, but the meta-predictor predicts which algorithm is the best for the next prediction.

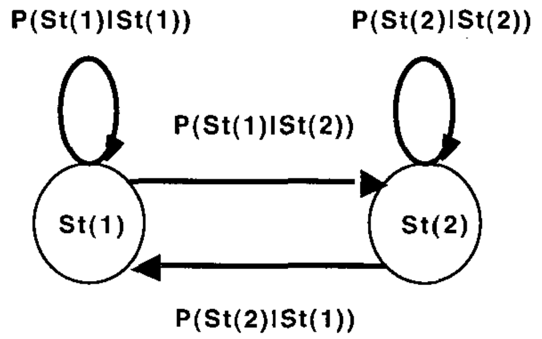

Figure 1: The Markov model used by meta-predictor.

Figure 1 is an illustration of the Markov model used by the meta-predictor at every prediction time. $St(1)$ means the state in which one predicting algorithm gave the most precise prediction for the last 2 prediction times, while $St(2)$ means the state in which no algorithm gave the most precise prediction in succession for the last 2 prediction times. $P(St(a) \mid St(b))$ is the probability of state transition from state a to state b.

Now, suppose one algorithm is selected that will calculate the most precise prediction for the next prediction time among prediction algorithms Mt_1, Mt_2, and Mt_3. If Mt_1 gives the most precise forecasts for last 2 prediction times, the current state is $St(1)$. For the next prediction, there are three possibilities: 1) Mt_1 gives the most precise prediction again, and the state is still $St(1)$ (let this probability be $p(Mt_1 \mid St(1) \rightarrow St(1))$), 2) Mt_2 gives the most precise prediction and the state changes to $St(2)$ (let this probability be $p(Mt_2 \mid St(1) \rightarrow St(2))$), and 3) Mt_3 gives the most precise prediction and the state changes to $St(2)$ (let this probability be $p(Mt_3 \mid St(1) \rightarrow St(2))$). The meta-predictor chooses the most probable of these 3 cases, comparing each probability, $p(Mt_1 \mid St(1) \rightarrow St(1))$, $p(Mt_2 \mid St(1) \rightarrow St(2))$, and $p(Mt_3 \mid St(1) \rightarrow St(2))$.

With this method, all predicting algorithms calculate predictions separately at every prediction time, meta-predictor obtains a measurement and recognizes which algorithm is the best. It then recalculates all the probabilities. Thus, the state transition matrix becomes accustomed to the environment as the meta-predictor predicts the load repeatedly. Moreover, even if there is one strongly precise algorithm, the state transition matrix is accustomed to mainly using the strong algorithm. As a result, this meta-predictor will not deteriorate the precision of each algorithm.

In the evaluation in this paper, we used the following algorithms for each prediction algorithm: mean value, median value, stochastic gradient, and an auto regressive model.

4. Network load prediction

4.1. Extended forecast of network load

There are several reasons for generating CPU load. However, packet transmission undoubtedly generates CPU load. Therefore, we group CPU load into two categories by causation: 1) load generated by applications without the network, and 2) load generated by tasks related to network interfaces, such as data transmission, communication with other nodes, and so on. In particular, the variance in CPU load 2) is assumed to directly concern network load variance. For this reason, in this paper, we called CPU load 2) "network-related CPU load," and applied it to network load prediction. Network-related CPU load can be forecasted in exactly the same way as CPU load forecast.

Therefore, after estimating and approximating seasonal variations in network-related CPU load, the approximated seasonal variations can be regarded as seasonal variations in network load variation. The variation in network load is strongly assumed to be similar to the variation in network-related CPU load. However, it is difficult to determine the expected network load measurement itself only through measurements of network-related CPU load. Thus, we calculate extended forecast of network load, $EP_{net}(t)$, using seasonal variations in network-related CPU load and the mean value of network load as the following formula.

$$EP_{net}(t) = M_{net} + FS_1(t) + \cdots + FS_N(t)$$

4.2. One-step-ahead prediction of network load

The same to as one-step-ahead CPU load predictions, one-step-ahead network load predictions are also required to be sufficiently precise and sensitive to resent complex change tendencies in the measurements. For this reason, we used the Markov model-based meta-predictor again to predict one-step-ahead network load measurement because we need to estimate irregular variation in network load, $I_{net}(t)$. Here, another meta-predictor is prepared only for network load prediction. The model itself is exactly the same as the model shown in Figure 1, but recalculation of probabilities is performed for CPU load and network load separately from the two different state transition models to allow the meta-predictor to be more sensitive to network load waves. As a result, one-step-ahead prediction of network load, $SP_{net}(t)$, is calculated by the following formula.

$$SP_{net}(t) = EP_{net}(t) + EI_{net}(t)$$

5. Evaluations

In this section, we evaluate our prediction method using CPU load and network load using Cisco 7513 in the Science and Engineering School of Waseda University by MRTG [19] for about 11 months (November 9, 2002 – October 14, 2003). The remainder of this section considers this taken CPU load to be the network-related CPU load of this router because this router is a special-purpose machine, and the taken CPU load is generated only by network-related task like routing. Cisco 7513 has several network interfaces. However, we used the network load of the FastEthernet interface, which connects the Science and Engineering School and the main campus of Waseda University.

If these experiments demonstrate that forecasts of network-related CPU load can be applied to network load forecasts, applying our forecast method to PC routers, PCs, workstations, and so on is well founded with the separation of network-related CPU load from the whole CPU load.

5.1. Evaluation of predicting network-related CPU load

Figure 2 shows the measurements of network-related CPU load and 8-day-long forecast of network-related CPU load at 3:00 on September 24, 2003. The measurements are the mean values of network-related CPU load every 2 hours. The accuracy of the measured values, such as the maximum value or the minimum value, need to be improved. However, the variation in load is well pictured.

Figure 3 shows the measurements and one-step-ahead predictions of network-related CPU load from 3:30 to 15:30 on October 14, 2003. The forecast was calculated at each monitoring time. The network-related CPU load was taken every 5 minutes. The prediction sometimes peaked slightly later than the measurement. However, the forecast pictures the maximum values and minimum values precisely, as well as the variation in the measurements. The mean average error rate, which is calculated using the following expression, was 6.2%.

$$\text{(Mean error rate)} = \frac{|\text{(measurement)} - \text{(prediction)}|}{(measurement)} \times 100[\%]$$

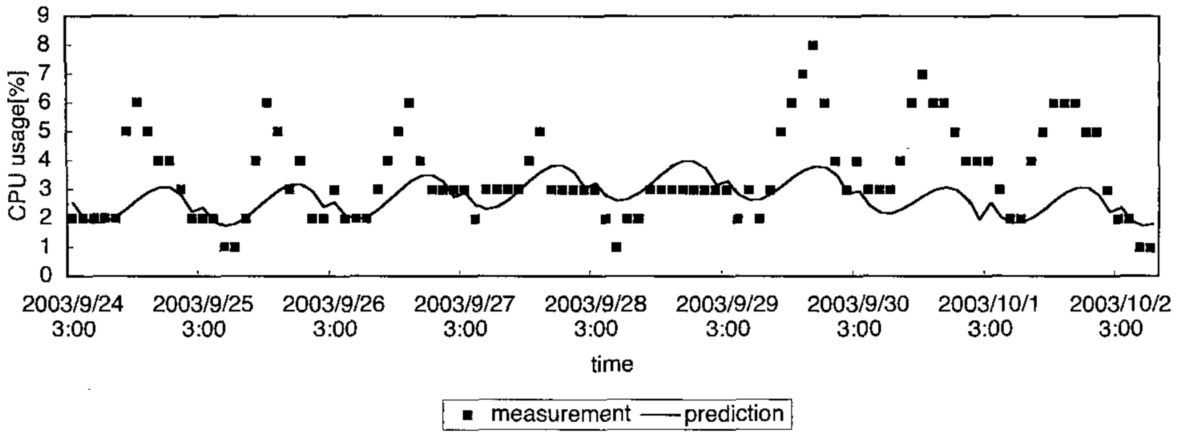

Figure 2: Measurements and extended forecast of network-related CPU load (September 24 – October 2, 2003).

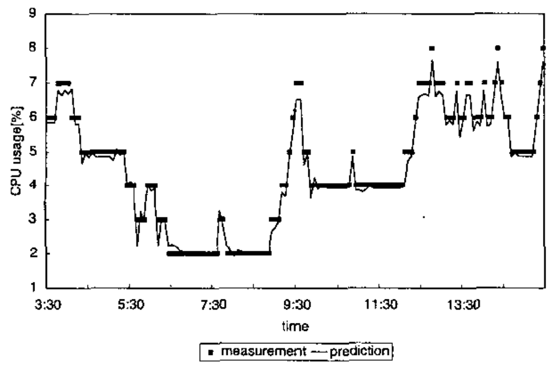

Figure 3: Measured values and one-step-ahead predictions of network-related CPU load (3:30 – 15:30 on October 14, 2003).

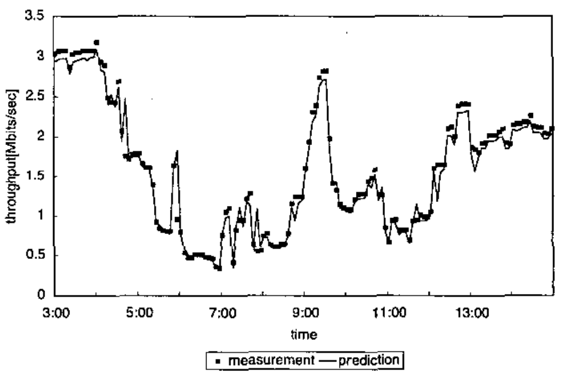

Figure 5: Measured values and one-step-ahead predictions of network load (3:00 – 15:00 on October 14, 2003).

5.2. Evaluation of prediction of network load

Figure 4 shows the measurements and 8-day-long forecast of network load at 23:00 on September 14, 2003. The measurements are the mean values of network load every 2 hours. The prediction sometimes peaked later than the measurements, but expressed the waves of the measurements well. Moreover, the forecast often provided maximum values and minimum values precisely. However, it is still necessary to improve precision.

Figure 5 shows the measurements and one-step-ahead predictions of network load from 3:00 to 15:00 on October 14, 2003. The predictions were calculated at each measuring time, and the measurements were taken every 5 minutes. The mean error rate was 9.4%. The forecasts picture complex network load variation well, as well as maximum values and minimum values. Sometimes the forecast peaked slightly later than the measurements, but the forecast did not continue to provide very inaccurate values.

Considering all the results of these experiments, network forecasting using seasonal variation in network-related CPU load is effective.

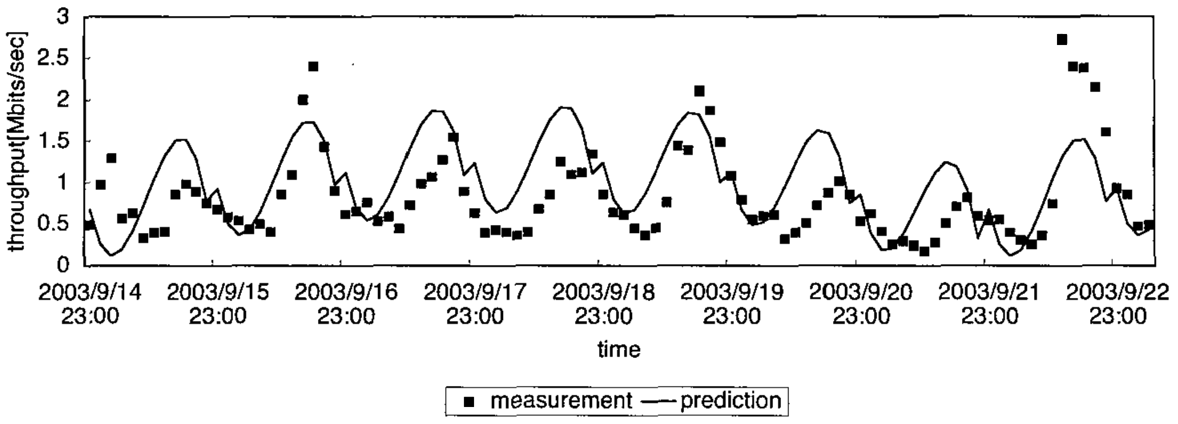

Figure 4: Measurements and extended forecast of network load (September 14 – 22, 2003).

Table 1: Results of evaluation of practicability of extended network-related CPU load forecast using the proposed method.

		Application starting time							
		0-24h	24-48h	48-72h	72-96h	96-120h	120-144h	144-168h	Total
Application running time	0-24h	12.9%	15.6%	9.6%	8.8%	5.8%	17.9%	12.8%	11.9%
	24- 48h	11.8%	7.0%	13.0%	6.2%	27.0%	19.1%	24.3%	15.5%
	48- 72h	4.1%	9.1%	15.4%	29.9%	22.7%	5.4%	9.1%	13.7%
	72- 96h	12.8%	21.3%	22.1%	10.5%	4.0%	11.1%	14.9%	13.8%
	96-120h	17.3%	18.0%	5.2%	4.2%	9.7%	22.6%	14.8%	13.1%
	120-144h	15.5%	8.8%	11.7%	15.0%	22.9%	13.4%	3.6%	13.0%
	144-168h	5.3%	13.6%	15.1%	18.7%	10.7%	5.9%	14.5%	12.0%
	Total	11.4%	13.4%	13.2%	13.3%	14.7%	13.6%	13.4%	13.3%

Table 2: Results of evaluation of practicability of extended network load forecast using the proposed method.

		Application starting time							
		0-24h	24-48h	48-72h	72-96h	96-120h	120-144h	144-168h	Total
Application running time	0-24h	48.0%	48.8%	14.1%	13.3%	36.3%	93.7%	20.0%	39.2%
	24-48h	30.6%	13.7%	12.1%	35.9%	81.0%	38.4%	35.3%	35.3%
	48-72h	24.9%	24.5%	38.6%	51.6%	38.7%	9.2%	12.7%	28.6%
	72-96h	39.1%	40.3%	34.1%	20.9%	10.3%	13.4%	10.9%	24.1%
	96-120h	38.5%	39.3%	12.0%	10.0%	13.5%	15.4%	12.1%	20.1%
	120-144h	31.0%	18.9%	5.5%	11.8%	13.3%	12.0%	23.2%	16.5%
	144-168h	10.4%	5.7%	14.3%	10.2%	12.5%	21.9%	18.7%	13.4%
	Total	31.8%	27.3%	18.7%	22.0%	29.4%	29.2%	19.0%	25.3%

5.3. Practicability

To confirm the practicability of our forecast, we conducted an experiment following the procedure below. This experiment assumed the arrangement of video conference or other network-related applications in need of predicting network throughput or CPU usage. The procedure of this experiment was: 1) generate a random number to decide when to start the application, 2) generate another random number to decide the execution time of the application, 3) calculate the expected mean load during the application runs using the proposed method, and 4) calculate the mean error rate against the real mean load.

Table 1 shows the results for network-related CPU load, and Table 2 shows the results for network load. The mean error rate for each case was the mean value of 1000 sets of the experiments. The predictions were calculated using only the extended forecast method, without the one-step-ahead prediction method. For example, the average error rate of network-related CPU load for applications, those application starting times vary from 24 hours to 48 hours and application running times vary from 48 hours to 72 hours, was 9.1%.

For this case, we could not find a clear relationship among application running time, application starting time, and forecast precision. However, the mean error rate of the prediction becomes smaller as the application run time becomes longer. This means that the proposed method forecasts the general curve very well, while the method sometimes predicts with emergent error. Moreover, when the predictions of network-related CPU load were with small error rate, the predictions of network load were generally also with small error rate. On the other hand, the predictions of network load were generally with large error rate when the predictions of network-related CPU load were with large error rate. Through these results, we confirmed that the load variance in the network-related CPU load strongly concerns network load variance. This means that using variance in network-related CPU load for predictions of network load is effective.

6. Conclusions

This paper proposes a method for extended forecast of CPU load and network load. The distinctive feature of our method is using seasonal variation for extended forecast, and using the Markov model-based meta-predictor in addition to seasonal variation for one-step-ahead prediction. Particularly for extended forecast of network load, we applied seasonal variation in network-related CPU load instead of seasonal variation in network load itself.

To confirm the accuracy and the practicability of our method, we conducted experiments using real CPU load and network load taken on a Cisco router. Throughout the experiments, our predictor expressed the expected curve of load variation well, giving very precise one-step-ahead prediction. The mean error rate for one-step-ahead prediction is 6.2% for CPU load, and 9.4% for network load.

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

References

[1] I. Foster and C. Kesselman, "The grid: blueprint for a new computing infrastructure," Morgan Kufmann, 1998.

[2] 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 Transactions on Parallel and Distributed Systems, vol.14, no.4, 2003.

[3] J. Basney and M. Livny, "Managing network resources in condor," Proc. 9th IEEE International Symposium on High Performance Distributed Computing (HPDC9), Pittsburgh, USA, 2000.

[4] S. Chapin, D. Katramatos, J. Karpovich, and A. Grimshaw, "The legion resource management system," Proc. 5th Workshop on Job Scheduling Strategies for Parallel Processing (JSSPP'99), San Juan, USA, 1999.

[5] R. Buyya, M. Murshed, and D. Abramson, "A deadline and budget constrained cost-time optimization algorithm for scheduling task farming applications on global grids," Proc. International Conference on Parallel and Distributed Processing Techniques and Application (PDPTA2002), Las Vegas, USA, 2002.

[6] H. Dail, O. Sievert, F. Berman, H. Casanova, A. YarKhan, S. Vadhiyar, J. Dongarra, C. Liu, D. Angulo, and I. Foster, "Scheduling in the grid application development software project," Resource Management in the Grid, Kluwer Publishers, 2003.

[7] M. Brune, J. Gehring, A. Keller, and A. Reinefeld, "Managing clusters of geographically distributed high-performance computers," Concurrency, Practice, and Experience, vol. II (15), pp.887-911, 1999.

[8] C. Liu, L. Yang, I. Foster and D. Angulo, "Design and evaluation of a resource selection framework for grid applications," Proc. 11th IEEE International Symposium on High Performance Distributed Computing (HPDC11), Edinburgh, Scotland, 2002.

[9] H. Dail, G. Obertelli, F. Berman, R. Wolski, and A. Grimshaw, "Application-aware scheduling of a magneto hydrodynamics application in the legion metasystem," Proc. 9th Heterogeneous Computing Workshop 2000, Cancun, Mexico, 2000.

[10] P.A. Dinda, "A prediction-based real-time scheduling advisor," Proc. 16th International Parallel and Distributed Processing Symposium (IPDPS 2002), Fort Lauderdale, USA, 2002.

[11] K. Czajkowski, I. Foster, N. Karonis, C. Kesselman, S. Martin, W.Smith, and S. Tuecke, "A resource management architecture for metacomputing systems," Proc. IPPS/SPDP '98 Workshop on Job Scheduling Strategies for Parallel Processing, Delta Orlando Resort, USA, 1998.

[12] M. Quinson, "Dynamic performance forecasting for network-enabled servers in a metacomputing environment," Proc. International Workshop on Performance Modeling, Evaluation, and Optimization of Parallel and Distributed Systems (PMEO-PDS'02), Fort Lauderdale, USA, 2002.

[13] M.O. McCracken, A. Snavely, and A. Malony, "Performance modeling for dynamic algorithm," Proc. Workshop on Performance Modeling (ICCS), Melbourne, Australia, 2003.

[14] R. Wolski, N. Spring, and J. Hayes, "The network weather service: a distributed resource performance forecasting service for metacomputing," Journal of Future Generation Computing Systems, vol.15, no.5-6, pp.757-768, 1999.

[15] R. Wolski, "Dynamically forecasting network performance using the network weather service," Journal of Cluster Computing, Vol.1, pp.119-132, 1998.
[16] P.A. Dinda, and D.R. O'Hallaron, "Host load prediction using linear models", Cluster Computing, 3, 2000.

[17] S. Akioka, and Y. Muraoka, "Predicting Network Load on Computational Grid," Proc. International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA2002), Las Vegas, USA, 2002.

[18] L. Yang, J.M. Schopf, and I. Foster, "Conservative scheduling: using predicted variance to improve scheduling decisions in dynamic environments," Proc. Supercomputing 2003, Phoenix, USA, 2003.

[19] "MRTG: the multi router traffic grapher," http://people.ee.ethz.ch/~oetiker/webtools/mrtg/.