Predicting the Performance of Wide Area Data Transfers

Sudharshan Vazhkudai [1,3] Jennifer M. Schopf [2,3] Ian Foster [3,4]
{vazhkuda, jms, foster}@mcs.anl.gov

[1] *Department of Computer and Information Sciences, The University of Mississippi*

[2] *Computer Science Department, Northwestern University*

[3] *Mathematics and Computer Sciences Division, Argonne National Laboratory*

[4] *Department of Computer Science, The University of Chicago*

Abstract

Distributed Data Grids may contain multiple copies of popular datasets, and users are interested in determining which replica can be accessed most efficiently. The answer to this question can depend on many factors, including physical characteristics, configurations, and current and future load on the various physical resources on the end-to-end path linking possible sources and sinks.

We propose an approach to this problem based on the use of predictive techniques. Specifically, we develop a predictive framework that combines integrated instrumentation to collect information about the end-to-end performance of past transfers, specialized predictors designed to synthesize predictions for future transfers from this information, and a data delivery infrastructure that provides users with access to predictions. We evaluate the performance of our predictors by applying them to some log data collected from a limited wide area testbed. These preliminary results provide some insights into the effectiveness of different predictors and suggest directions for future research.

Keywords: *Grids, data transfer prediction, replica selection, information services.*

1 Introduction

As Grid Computing, the coordinated use of distributed resources, becomes more commonplace, the resource usage questions posed by users are changing. Many recent applications use Grid systems as distributed data stores [7, 15, 23, 24, 26, 31]. For example, various high-energy physics experiments have agreed on a tiered Data Grid architecture [21, 20] in which all data (approximately 20 petabytes by 2006) is located at a single Tier 0 site; various (overlapping) subsets of this data are located at national Tier 1 sites, each with roughly one tenth the capacity; yet smaller subsets are cached at smaller regional Tier 2 regional sites; and so on. Therefore a particular data set is likely to be located at multiple sites.

Different sites may have different performance characteristics, due to different storage system architectures, network connectivity, and load. Thus, users (or brokers acting on their behalf) may want to be able to determine the site from which particular data sets can be retrieved most efficiently, especially as data sets of interest tend to be large (1-1000 MB). It is this *replica selection* problem that we address in this paper.

One approach to replica selection is to construct performance models [28] for each system component (storage systems, networks, clients) and then use these models to determine an optimal schedule for all data transfers. This is an approach widely used in classical scheduling, where the resources are typically CPUs not network components, and the action is running a task, not transferring a file [30, 36]. However, in practice, system components are neither dedicated to our use nor under our full control, and so their behavior can vary in unpredictable ways as a result of competing traffic and load, un-modeled interactions, or the lack of available data.

A promising alternative approach to system characterization is to use observations of past application performance of the entire system, not on a component-by-component level, is to construct predictions of future performance. For example, this technique is used by the Network Weather Service (NWS) [34] or NetLogger [25] to predict network and CPU behavior, and by Downey [9] and Smith [29] to predict queue wait times. The use of whole system observation has three relevant properties for our usage. First, we can construct such predictions without detailed knowledge of the underlying physical devices. Second, these predictions can, in principle, capture both evolution in system configuration and temporal patterns in load. Third, such predictions are for end-to-end behavior, which is typically what is of interest to the user.

These considerations motivate us to apply predictive techniques to the problem of estimating data transfer times between a storage system and a remote client. This is a more challenging problem than predicting just network performance because it includes issues associated with storage systems that are less amenable to "law of large numbers" arguments than wide area networks in that it is no longer the case that one additional flow or task has an insignificant effect on overall performance. Nevertheless, we show some encouraging early results.

A performance prediction system must address two primary issues: obtaining the measurements used to make predictions, and making predictions based on those measurements. There are obvious tradeoffs between measurement overhead and predictive accuracy. We must also address the issue of how to deliver predictions to the user.

We approach these problems as follows. We obtain measurements by instrumenting a high-performance data server (specifically, a GridFTP [1] file transfer service) to record performance information for each file transfer that it performs. We construct predictions by applying various predictors, described below, to this data. We deliver predictions to the user via an information provider that uses the Globus Toolkit's Meta Directory Service (MDS) protocols [6] to notify others of its existence and respond to queries.

In the remainder of this paper, we describe these three aspects of our system and then present the results of experimental studies in which we evaluate its accuracy. We conclude with a discussion of related and planned future work.

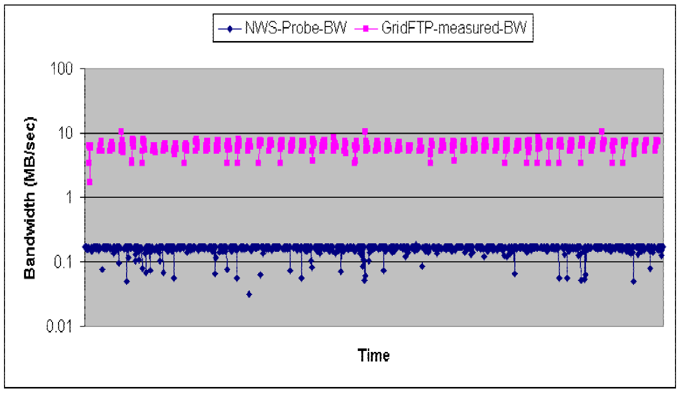

Figure 1: ISI-ANL GridFTP end-to-end bandwidth and NWS probe bandwidth

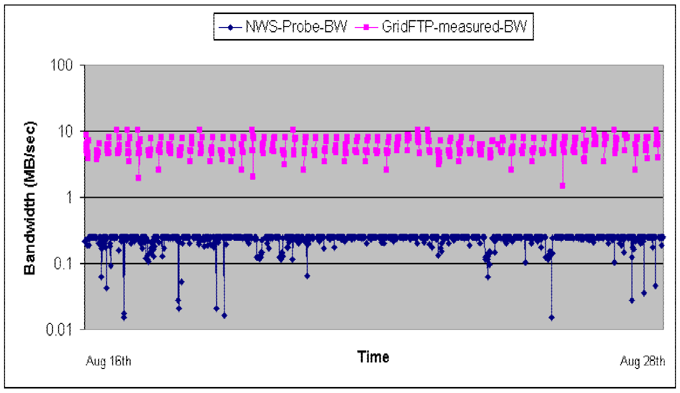

Figure 2: LBL-ANL GridFTP end-to-end bandwidth and NWS probe bandwidth

2 Monitoring GridFTP Performance

Our goal is to be able to obtain an accurate prediction of the time required to transfer a file between a storage system S and a client C—either now, or at some point in the future. This goal is challenging because of the number of devices involved in the end-to-end path between S and C, and because, in the absence of any "insider" knowledge, the future performance of each device may vary in unpredictable ways.

One approach to simplifying this problem is to assume that one component of the end-to-end path, for example the wide area network, is the rate-limiting step, such as it was in the case of our experiments. We can then use techniques for predicting the performance of that component (for example, NWS for wide area networks) to obtain approximate predictions for the end-to-end path.

Source IP	File Name	File Size (Bytes)	Volume	StartTime (Timestamp)	EndTime (Timestamp)	TotalTime (Seconds)	Bandwidth (KB/Sec)	Read/Write	Streams	TCP-Buffer
140.221.65.69	/home/ftp/vazhkuda/10MB	10240000	/home/ftp	998988165	998988169	4	2560	Read	8	1000000
140.221.65.69	/home/ftp/vazhkuda/25MB	25600000	/home/ftp	998988172	998988176	4	6400	Read	8	1000000
140.221.65.69	/home/ftp/vazhkuda/50MB	51200000	/home/ftp	998988181	998988190	9	5688	Read	8	1000000
140.221.65.69	/home/ftp/vazhkuda/100MB	102400000	/home/ftp	998988199	998988221	22	4654	Read	8	1000000
140.221.65.69	/home/ftp/vazhkuda/250MB	256000000	/home/ftp	998988224	998988256	33	8000	Read	8	1000000
140.221.65.69	/home/ftp/vazhkuda/500MB	512000000	/home/ftp	998988258	998988335	67	7641	Read	8	1000000
140.221.65.69	/home/ftp/vazhkuda/750MB	768000000	/home/ftp	998988338	998988425	97	7917	Read	8	1000000
140.221.65.69	/home/ftp/vazhkuda/1GB	1024000000	/home/ftp	998988428	998988554	126	8126	Read	8	1000000

Table 1: Shows a sample set from a log of file transfers between Argonne National Lab and Lawrence Berkeley National Lab. The bandwidth values logged are sustained measures through the transfer. The end-to-end GridFTP bandwidth is obtained by the formula: *BW = Filesize / TransferTime*

However, we find that in practice such approximations have only limited value for two reasons. First, the rapidly increasing performance of wide area links means that they are often not the rate-limiting step in a data transfer. Second, the techniques used to estimate performance in systems such as NWS (which by default uses 64 KB probes and default TCP buffer sizes) are a poor approximation to the network traffic associated with the large file sizes and possible parallelism associated with typical GridFTP transfers.

Figures 1 and 2 illustrate the differences in performance that can arise due to these issues. These two figures plot (in logarithmic scale) the performance measured in a set of controlled experiments with NWS probes and GridFTP transfers over a two-week period between two pairs of sites. Each figure shows approximately 1500 NWS probes, conducted every 5 minutes, and approximately 400 GridFTP transfers at irregular intervals.

The NWS measurements indicate network bandwidth to be well under 1 MB/Sec, while end-to-end GridFTP had a significantly higher transfer rate. More problematic, however, we see considerably greater variability in the GridFTP measures, ranging from 3 to 8MB/sec in both cases, showing that simple data transformations will not improve its predictive merits for this application. The NWS measurements are not the right tool to use, quantitatively or qualitatively, for accurate estimates of GridFTP costs.

These results suggest that we need to obtain measurements of the entire transfer function, not just the transport. To this end, we instrument a GridFTP server [5] to record the performance achieved for every data transfer that it performs. This information, along with metadata indicating the nature of the transfer, serves as the input to our predictors. This use of actual data transfers to collect performance information, rather than synthetic probes, can give us more accurate data for the full function and does not impose any additional probing overhead. The downside of this is that we have no control over the intervals at which data is collected. This can limit the predictive techniques that apply to the data, as detailed in Section 3. In principle, our system could be extended to perform file transfer probes as well, but we do not consider that approach here.

2.1 Instrumenting GridFTP

GridFTP is part of the Globus Toolkit [11] and is widely used as a secure, high-performance data transfer protocol [2, 5, 7, 11, 15, 27]. It extends standard FTP with several features needed in Grid environments, such as security on control and data channels, multiple data channels for parallel transfers, partial file transfers, and third party transfers [1]. GridFTP essential consists of two modules, the control module and the client module. The control module is responsible for managing connection, authentication, creation of control and data channels and reading and writing data. Separate control and data channels facilitate parallel transfers. The client module is responsible for higher-level operations such as file get and put operations or partial transfers.

We instrumented GridFTP by adding mechanisms to log performance information for every file transfer. We do not add any capabilities to GridFTP itself, we merely record the data and time the action took to complete. Log entries include source address, file name, file size, number of parallel streams, tcp buffer size for the transfer, start and end timestamps, total time consumed by the transfer, aggregate bandwidth achieved for the transfer, nature of the operation (read or write), and logical volume to/from which file was transferred. A sample log is shown in Table 1. Logging is achieved by timing the entire GridFTP transfer mechanism. Certain log entries are collected specifically, while others are derived from GridFTP resources.

The monitoring code is non-intrusive. The majority of the overhead of the monitor is in the timing routines, with a smaller percentage spent gathering the information mentioned above and performing a write operation. The entire logging process consumes on average approximately 25 milliseconds per transfer, which is insignificant when compared to the total transfer time.

We log data to a standard location in the file system hierarchy, with a single log file used for all transfers made to/from the server. Entries are logged in the Universal Logging Format (ULM [32]) "Keyword=Value" format. Each log entry is well under 512 bytes.

Transfer logs can grow quickly in size in a busy site. As old data has less relevance to predictions, we can trim logs based on a running window, as is done in the Network Weather Service, although we have not yet implemented this strategy. An alternate strategy used by NetLogger [25] is to flush the logs to persistent storage (either disk or network) and restart logging. We are exploring these strategies for GridFTP logs.

3 Prediction Techniques

Simply collecting the data from the GridFTP monitor is not sufficient to make a replica selection decision. For this information to be useful in most cases a prediction of future behavior is needed not just a recitation of past behavior. In this section we briefly categorize possible approaches, their pros and cons, and detail our specific implementations.

Predictive techniques can be categorized along several different independent axes, of which we consider three: basic mathematical technique, context-insensitive filtering (using only the last 5 measurements, for example), and context-sensitive filtering (for example choosing only to use data for similarly sized file transfers). We detail these techniques below, and then describe the initial set of predictors we use for our data.

3. 1 Basic Mathematical Functions

Mathematical functions for predictions are generally grouped into three categories: mean-based, median-based, and auto-reduction techniques.

Mean-based, or averaging, techniques are a standard class of predictors that use arithmetic averaging (as an estimate of the mean value) over some portion of the measurement history to provide an estimate of future behavior. The general formula for these techniques is the sum of the previous "n" values over the number of measurements.

Mean-based predictors vary with the amount of history information used in their calculations and the amount of weight put on each value. For example, a total average uses the entire set of history data with each value weighted equally, but if more recent behavior has better predictive value, then a subset of the data is used.

A second class of standard predictors is based around evaluating the median of a set of values. The standard way to calculate this is given an ordered list of t values, if t is odd, the median is the (t+1)/2 value, if t is even, the median is half of the t/2 value added with the (t+1)/2 value.

Median-based predictors are particularly useful if the measurements contain randomly occurring asymmetric outliers that are rejected. However, they lack some of the smoothing that occurs with a mean-based method, possibly resulting in forecasts with a considerable amount of jitter [17].

A third class of common predictors is Autoregressive Models [34, 14, 17]. These have been used by other predictive techniques, such as NWS. We used both mean-based and average-based predictors for our predictions.

3.2 Context-Insensitive Factors

More recent values are often better predictors of future behavior than an entire data set, no matter which mathematical technique is used to calculate a prediction. Because of this, there are many different variants in selecting a set of recent measurements to use in a prediction calculation.

The fixed-length, or sliding window, average is calculated using only a set number of previous measurements to calculate the average. The number of measurements can be chosen statically or dynamically depending on the system. We use only static selection techniques in this work, although others choose dynamically. Further options for dynamically selecting window size are discussed in [34].

The degenerative case of this strategy is when only the last measurement is used to predict the future behavior. Work by Downey and Harchol-Balter [19] shows that this is a useful predictor for CPU resources, for example.

Instead of selecting the number of recent measurements to use in a prediction we also consider using only a set of measurements from a previous window of time. Unlike other systems where measurements are taken at regular intervals [8, 34] we make this distinction because our measurements can be spaced irregularly in time. The lack of regular samples means that this approach can reflect trends more accurately than selecting a specific number of previous measurements since it uses temporal windows that capture recent fluctuations thereby helping to ensure that recent, hopefully more predictive, data is used.

Much as the number of measurements to be included in a prediction could be selected dynamically, the window of time used could be decided dynamically, perhaps based on the load of a server—busy sites could use a smaller time frame while still having enough values for statistically significant decisions.

In our predictions, we use methods that select measurement sets of the last 5, 15 and 25 measurements, as well as time-based sets of the last 5 hours, 15 hours, and 25 hours.

3.3 Context-Sensitive Factors

It is often the case that filtering a data set to eliminate non-related values can result in a more accurate prediction. For example, a prediction of salary is more accurate when factors such as previous training, education, and years at the position are used to limit the data set of interest.

In the case of the GridFTP monitoring data, initial results showed that file transfer rates had a strong correlation to file size. Studies of Internet traffic have also revealed that small files achieve low bandwidths while larger files tend to have high bandwidths [4, 16]. This is thought to be primarily due to the startup overhead associated with the TCP slow start mechanism that probes the bandwidth at connection startup. Recent work has attempted to perform class-based isolation of TCP flows [35] or startup optimizations [37, 38] to attempt to mitigate this problem. As a proof of concept, we found 5-10 percent improvement on average when using file-size classification over when the entire history file was used to calculate a prediction. Graphs are presented in later sections to support these claims.

For our GridFTP transfer data we ran a series of tests between our testbed sites to try to categorize the data sizes into a small number of classes. It should be noted that the classes we used apply to the set of hosts for our testbed and further work is needed to generalize this notion. We categorized our data into four sets: 0-50 MB, 50-250 MB, 250-750 MB and more than 750MB based on the achievable bandwidth.

3.4 Predictors Used

In our initial experiments presented in Section 5 we used a set of 24 predictors over our data sets, twelve predictors each over the entire data set ignoring the context-sensitive factor of data-transfer size, and the same twelve using previous data partitioned by file size. These predictors are summarized in Table 2. As noted earlier, we have primarily considered mean-based and median-based predictors for our work. Of course, many other variants for predictors are possible [8, 29, 34]. We plan to examine these in future work. Also, rather than choosing just a single prediction technique, we could also evaluate a number of them and choose the most appropriate one on the fly, as is done by NWS. We are considering this strategy for future work.

	Average based	Median based
All data	AVG	MED
Last 5 Measures	AVG5	MED5
Last 15 Measures	AVG15	MED15
Last 25 Measures	AVG25	MED25
Last 5 hour	AVG5hr	
Last 15 hour	AVG15hr	
Last 25 hour	AVG25hr	
Last value	LV	

Table 2: Classes of Predictors

4 Grid Information Infrastructure

Gathering the data is just the first step in building a service to provide predictions for replica selection [5, 33,]. The second step, discussed in Section 3, is making a

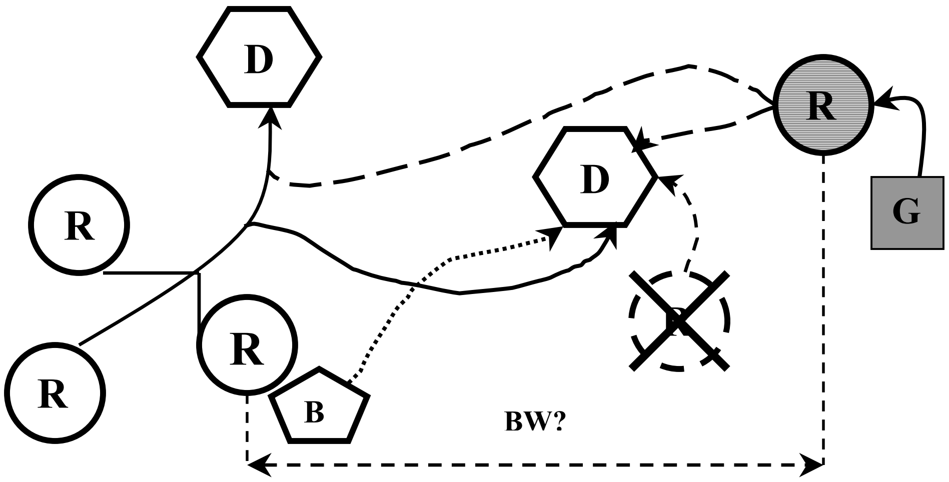

Figure 3: Directory Services with GIIS (D), GRIS (R), GridFTP Information Provider (G) and Broker (B). Depicts GRIS registration, GIIS functionality with disconnected GRIS and broker enquiries to GIIS on performance information.

prediction of future behavior based on past information. The third step, described in this section, is defining object classes, integrating this information with a resource provider, and then allowing this information to be discovered in the context of an information service. These last two issues are addressed in the Globus information service infrastructure with a GRIS and a GIIS, described below.

Within Globus, the information infrastructure is handled by MDS-2 [6]. This service provides a configurable information provider component called a Grid Resource Information Service (GRIS) and a configurable aggregate directory component called a Grid Index Information Service (GIIS) (Figure 3).

These components interact with each other and higher-level services (or users) using two basic protocols: a soft-state registration protocol for identifying entities participating in the information service, and an inquiry protocol for retrieval of information about those entities. In brief, a GRIS uses the registration protocol to notify a GIIS (or other higher-level service) of its existence; a GIIS uses the inquiry protocol to obtain information from the GRIS (or a GIIS lower in the hierarchy) known to that provider, which it merges into an aggregate view [6].

A GRIS communicates with an information provider via a well-defined API. To control the intrusiveness of GRIS operation, improve response time, and maximize deployment flexibility, each provider's results may be cached for a configurable period of time to reduce the number of provider invocations; this cache time-to-live (TTL) is specified per-provider as part of the local configuration. GRIS servers use the Light Weight Directory Access Protocol (LDAP) [22], and publish information in LDIF, categorized under different object classes (comprising of multiple attributes and their associated values). Information is easily accessible when published in attribute-value pairs and LDIF provides a mechanism to do so.

4.1 GridFTP Information Provider

For the GridFTP monitoring data, we built an information provider that accesses the log data to advertise a set of recent measurements as well as some summary statistic data. To generate statistical information on transfers we developed LDAP shell-backend scripts to filter the information in the logs resulting in average, maximum, and minimum bandwidth value estimates. In addition, we developed schemas [13, 33] for this data.

Table 3 presents a fragment of the output from a GridFTP information provider. Combined, these provide a GridFTP performance information provider to process logs, by building schemas and scripts to publish statistical information. This information is published using the GRIS server on each storage server (replica site) that runs GridFTP. Each GRIS, comprising the GridFTP information provider, is registered with a GIIS for the testbed, so the index server can be populated with performance information from the replica servers (Figure 3).

From our preliminary controlled experiments, a log of approximately 100KB (comprising transfers from several sites—around 700 log entries) took the information provider approximately 1 to 2 seconds to filter the logs, classify the entries into object classes, obtain averages and compute predictions. The information provider delivers details for every client that has made transfers to the server. Further, this information is provided on a per-server-volume basis (Table 3).

4.2 Use Case

Figure 3 depicts aggregated directories (D) and individual information servers (R) registered with them. Let us consider the case where a new GRIS, R, with the GridFTP information provider, G, attempts to register with the GIIS, D.

Sample LDIF output from GridFTP Performance Information Provider

```
dn:"client_ip=140.221.65.69,volume=/home/ftp,
hostname=dpsslx04.lbl.gov,dc=lbl,dc=gov,o=grid"
cn:"140.221.65.69"
hostname:"dpsslx04.lbl.gov"
gridftpurl:"gsiftp://dpsslx04.lbl.gov:61000"
volume:"/home/ftp"
minrdbandwidth:1462K
maxrdbandwidth:12800K
avgrdbandwidth:6062K
avgrdbandwidthtenmbrange:5714K
avgrdbandwidthhunmbrange:5363K
avgrdbandwidthfivehunmbrange:6426K
avgrdbandwidthgigrange:7754K
medianrdbandwidth:5807K
………………………………
```

Table 3: A fragment of the output from the GridFTP information provider registered with the GRIS at LBL. The output includes minimum, maximum, and average bandwidths for various file size categories; median bandwidth; and the location of the GridFTP server at LBL.

Typically, the GridFTP information provider is launched by the GRIS at each site in response to enquiries from the index servers (GIIS) or a broker. A sample broker query depicted in Figure 3, attempts to find the bandwidth prediction between two sites. This is done by having the GIIS search each registered GRIS, and then caching the results. Subsequent searches are returned from the cache at the GIIS (if the time to live value indicates the cached copy is current).

The GRIS sites for our usage are servers hosting replicas, one of which is selected by brokers acting on behalf of client applications. In response to these queries, the information specified in Table 3 is returned. A piece of this information is the prediction returned by the information provider.

5 Experimental Results

In this section, we evaluate the twenty-four predictors described in Section 3 on log files obtained from GridFTP transfers on a test bed of three sites, Argonne National Laboratory (ANL), the University of Southern California's Information Sciences Institute (ISI) and Lawrence Berkeley Laboratory (LBL).

$$\textbf{\% Error} = \frac{(\,|\,\mathit{Measured}_{BW} - \mathit{Predicted}_{BW}\,|\,)}{\mathit{Measured}_{BW}} \; \textbf{* 100}$$

Equation 1: Percent error estimate is the ratio of the absolute value of the difference between measured and predicted bandwidths over the measured bandwidth

5.1 Log files, Pre-processing, and Error Calculations

The log files we use in this section come from two GridFTP log files, containing transfer data collected over a two-week period during the summer of 2001 over two wide area links: LBL to ANL and ISI to ANL.

Each log file contains approximately 400 to 450 transfers collected over a two-week period. The logs were generated using controlled GridFTP experiments that were performed everyday from 6pm to 8am CDT, selecting a random file size from the set {1M, 2M, 5M, 25M, 50M, 100M, 150M, 250M, 400M, 500M, 750M, 1G} and randomly sleeping from 1 minute to 10 hours between file transfers. Tables 2 and 4 show how many values were obtained for each file classification size (as discussed in Section 3.3). Traces of log data can be obtained from [12]

For each data set and predictor, we use a 15-value training set, that is, we assumed that at the start of a predictive technique there were at least 15 values in the

		A	A5	A15	A25	M	M5	M15	M25	A5hr	A15hr	A25hr	LV
ALL	LBL	26.0	28.0	25.2	25.3	21.8	27.5	22.5	22.7	27.2	25.0	25.5	25.2
	ISI	16.0	18.0	15.7	15.2	15.8	19.1	16.4	15.4	20.2	16.0	15.8	17.0
10MB	LBL	24.9	25.2	24.7	23.6	19.3	20.6	21.7	18.9	33	15.7	16.7	20.6
	ISI	18.6	21.2	18.3	18.4	19	20.8	16.3	18.4	25.2	19.4	18.9	27.3
100MB	LBL	16.2	15.3	17.4	18.9	15.3	14.4	17.2	16.2	15.0	15.7	16.7	16.1
	ISI	10.5	9.7	8.7	8.9	10.0	9.8	9.0	9.1	11.3	11.1	11.0	11.5
500MB	LBL	21.6	21.3	21.0	22.8	22.2	23.1	22.2	24.4	23	20.4	20.3	26.5
	ISI	7.2	5.3	5.3	6.6	6.9	5.59	5.4	6.2	5.8	5.3	5.5	6.5
1GB	LBL	5.8	5.6	5.2	8.0	5.0	5.3	5.4	6.8	4.8	6.2	5.8	4.8
	ISI	8.9	6.6	9.9	9.4	9.3	6.5	10.3	9.7	7.9	6.9	7.2	7.9

Table 4: Percent error rates for all predictors for transfers from LBL-ANL and ISI-ANL

log file. This does not imply, in the case of using context sensitive information, that there were 15 relevant values, only that there were 15 values in the logs to begin to work with.

5.2 Summary of Results

We compare our predicted transfer rates to the actual transfer rates, as shown in Figures 4 to 18. In addition, we summarize the accurateness of our predictions using the percentage error calculation, shown in Equation 1, as summarized in Table 4.

Figures 4 through 9 show the predictions and actual values for our 12 predictors without using the context-sensitive information of file-size sorting, 4, 5, and 6 for LBL-ANL, and 7, 8, and9 for ISI-ANL. Figures 10 through 15 show predictors with context sensitive grouping for LBL-ANL and ISI-ANL.

The major result from these predictions is that even simple techniques are at worst off by about 25%, quite respectable for pragmatic prediction systems. More specifically, we see a marked improvement in predictions when we sort the data by file size and it's at least 100 MB in size (Figures 16 and 17 compare error rates of predictors in the context sensitive and insensitive cases), and in general, large file sizes transfers seem to be more predictable than smaller file transfers.

For our data sets, we did not see a noticeable advantage for limiting either average or median techniques by sliding window or time frames, but this is likely due to the controlled experimental nature of our data.

We are currently running further experiments to take our proof-of-concept system to production systems in general. Even if those results are inconclusive as to which predictor is most efficient, using a dynamically selected predictor, such as is done in the NWS system, would give users access to good predictions for replica selection.

6 Conclusions and Future Work

In this paper we have described a technique that takes a step towards resolving the file replica selection problem. We detailed a monitor for GridFTP file transfer behavior, several possible predictive techniques, and how data related to this is made accessible as part of the Globus information service, MDS, by means of an information provider that uses GRIS/GIIS components.

Since our work with predictions was inconclusive, our future work will include using additional prediction techniques, such as those used in the NWS, as well as the possibility of using the NWS dynamic selection techniques.

In addition, we plan to investigate using both basic predictions on the sporadic data combined with more regular NWS measurements and predictions for small regular data movement to overcome the drawbacks of each approach in isolation.

In addition, to extend the usability of these approaches, we plan to experiment with techniques that will let us extrapolate data when there is no previous transfer data between two sites [10], or to leverage off of other available data in these settings.

Acknowledgements

Many thanks to the Globus team, especially Bill Allcock. This work was supported in part by the Mathematical

Information and Computational Sciences Division Subprogram of the Office of Advanced Scientific Computing Research, U.S. Department of Energy, under contract W-31-109-Eng-38, and by NSF Awards ACI-00933000 and PACI#763-04/ACI-9619019.

References

[1] W. Allcock, J. Bester, J. Bresnahan, A. Chervenak, I. Foster, C. Kesselman, S. Meder, V. Nefedova, D. Quesnel, S. Tuecke. Secure, Efficient Data Transport and Replica Management for High-Performance Data-Intensive Computing. (*Submitted to IEEE Mass Storage Conference*, April 2001).

[2] W. Allcock, I. Foster, V. Nefedova, A. Chevrenak, E. Deelman, C. Kesselman, A. Sim, A. Shoshani, B. Drach, D. Williams, High-Performance Remote Access to Climate Simulation Data: A Challenge Problem for Data Grid Technologies, *To be published in the Proceedings of Supercomputing (SC'01),* 2001.

[3] S. Basu, A. Mukherjee and S. Kilvansky, Time series models for internet traffic, Tech rep GIT-CC-95-27, Georgia institute of technology, 1996.

[4] N. Cardwell, S. Savage, and T. Anderson, Modeling the performance of short TCP connections, Technical Report, Computer Science Department, Washington University, Nov. 1998.

[5] A. Chervenak, I. Foster, C. Kesselman, C. Salisbury, S. Tuecke, The Data Grid: Towards an Architecture for the Distributed Management and Analysis of Large Scientific Datasets. (To be published in the *Journal of Network and Computer Applications*).

[6] K. Czajkowski, S. Fitzgerald, I. Foster, C. Kesselman, Grid Information Services for Distributed Resource Sharing. *Proceedings of the Tenth IEEE International Symposium on High-Performance Distributed Computing (HPDC-10)*, IEEE Press, August 2001.

[7] The Data Grid Project, http://www.eu-datagrid.org

[8] P. Dinda, D. O'Hallaron, Host Load Prediction Using Linear Models*, Cluster Computing*, Volume 3, Number 4, 2000.

[9] A. Downey, Queue Times on Space-Sharing Parallel Computers, *Proceedings of the 11th International Parallel Processing Symposium*, 1997.

[10] M. Faerman, Alan Su, Rich Wolski *and* Francine Berman, Adaptive Performance Prediction for Distributed Data-Intensive Applications. *Proceedings of the ACM/IEEE SC99 Conference on High Performance Networking and Computing*, Portland, Oregon, November, 1999.

[11] I. Foster, C. Kesselman, The Globus Project: A Status Report. *Proc. IPPS/SPDP '98 Heterogeneous Computing Workshop*, pg. 4-18, 1998.

[12] GridFTP predictor Trace Data, http://www.mcs.anl.gov/~vazhkuda/Traces

[13] GridFTP Information Provider Schema, http://www.mcs.anl.gov/~vazhkuda/Schema

[14] N. Groschwitz and G. Polyzos, A time series model of long-term traffic on the nsfnet backbone. *In Proceedings of the IEEE conference on communications (ICC'94)*, May, 1994.

[15] The GriPhyN Project, http://www.griphyn.org

[16] L. Guo and I. Matta, The War Between Mice and Elephants, Technical Report BU-CS-2001-005, Boston University, Computer Science Department, May 2001.

[17] R. Haddad and T. Parsons, Digital Signal Processing: Theory, Applications, and Hardware. *Computer Science Press*, 1991.

[18] M. Hafeez, A. Samar, H. Stockinger, A Data Grid Prototype for Distributed Data Production in CMS,*VII International Workshop on Advanced Computing and Analysis Techniques in Physics Research (ACAT2000),* October 2000.

[19] M. Harchol-balter, A. Downey, Exploiting process lifetime distributions for dynamic load balancing, *In Proceedings of the 1996 Sigmetrics Conference on Measurement and Modeling of Computer Systems*, 1996.

[20] K. Holtman, Object Level Replication for Physics. *Proceedings of 4th Annual Globus Retreat*, Pittsburgh, July 2000.

[21] W. Hoschek, J. Jaen-Martinez, A. Samar, H. Stockinger, Data Management in an International Grid Project. *2000 International Workshop on Grid Computing (GRID 2000)*, Bangalore, India, December 2000.

[22] T.A. Howes and M.C. Smith, *LDAP Programming Directory-Enabled Application with Lightweight Directory Access Protocol.* Technology Series, MacMillan, 1997.

[23] The LIGO Experiment, http://www.ligo.caltech.edu/

[24] D. Malon, E. May, S. Resconi, J. Shank, A. Vaniachine, T. Wenaus, S. Youssef, Grid-enabled Data Access in the ATLAS Athena Framework, *Proceedings of Computing and High Energety Physics 2001 (CHEP'01) Conference*, 2001.

[25] NetLogger: A Methodology for Monitoring and Analysis of Distributed Systems. http://www-didc.lbl.gov/NetLogger

[26] H. Newman, R. Mount, The Particle Physics Data Grid, www.cacr.caltech.edu/ppdg

[27] A. Samar, H. Stockinger. Grid Data Danagement Pilot (GDMP): A Tool for Wide Area Replication, *IASTED International Conference on Applied Informatics (AI2001)*, Innsbruck, Austria, February 2001.

[28] X. Shen and A. Choudhary, A Multi-Storage Resource Architecture and I/O, Performance Prediction for Scientific Computing. *Proceedings of the 9th IEEE Symposium on High Performance Distributed Computing,* pages 21-30. IEEE-Press, 2000.

[29] W. Smith, I. Foster, and V. Taylor, Predicting Application Run Times Using Historical Information. *In Proceedings of the IPPS/SPDP '98 Workshop on Job Scheduling Strategies for Parallel Processing*, 1998.

[30] A. Thomasian, P. F. Bay, Analysis Queuing Network Models for Parallel Processing of Task Systems. *IEEE Transactions on Computers c-35,* vol 12, December 1986.

[31] I. Terekhov, R. Pordes, V. White, L. Lueking, L. Carpenter, H. Schellman, J. Trumbo, S. Veseli, M. Vranicar, Distributed Data Access and Resource Management in the D0 SAM System. *Proceedings of HPDC 2000,* SanFrancisco, August, 2000.

[32] Universal Format for Logger Messages, http://www-didc.lbl.gov/NetLogger/draft-abela-ulm-05.txt

[33] S. Vazhkudai, S. Tuecke, I. Foster, Replica Selection in the Globus Data Grid. *Proceedings of the First IEEE/ACM International Conference on Cluster Computing and the Grid (CCGRID 2001)*, Pages 106-113, IEEE Computer Society Press, May 2001.

[34] R. Wolski, Dynamically Forecasting Network Performance Using the Network Weather Service. *Journal of Cluster Computing*, 1998.

[35] S. Yilmaz, I. Matta, On Class-based Isolation of UDP, Short-lived and Long-lived TCP Flows, Technical Report BU-CS-2001-011, Boston University, Computer Science Department, June 2001.

[36] M.J. Zaki, W. Li, S. Parthasarathy, Customized Dynamic Load Balancing for Network of Workstations. *Proceedings of HPDC'96*, 1996.

[37] Y. Zhang, L. Qiu, and S. Keshav, Speeding Up Short Data Transfers: Theory, Architecture Support, and Simulation Results, *In Proc. NOSSDAV 2000*, Chapel Hill, NC, June 2000.

[38] Y. Zhang and L. Qiu and S. Keshav, Optimizing {TCP} Start-up Performance, Technical Report TR99-1731, Cornell University, Department of Computer Science, 1999.

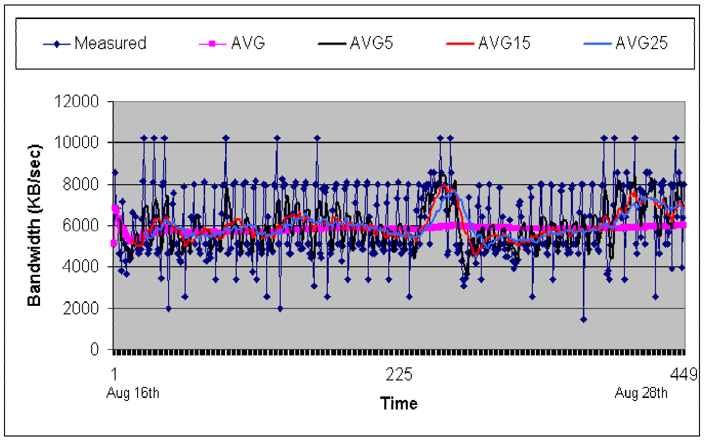

Figure 4: LBL-ANL all transfers and mean-based predictors. Depicts how window based approaches fluctuate and capture variations while how simple AVG does not vary.

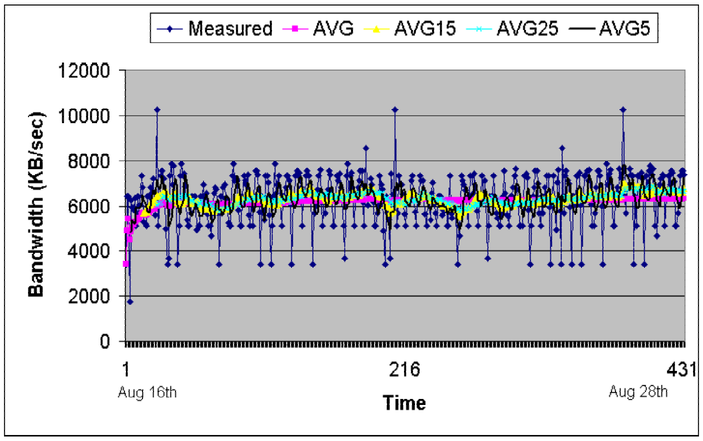

Figure 7: ISI-ANL all transfers and mean-based predictors

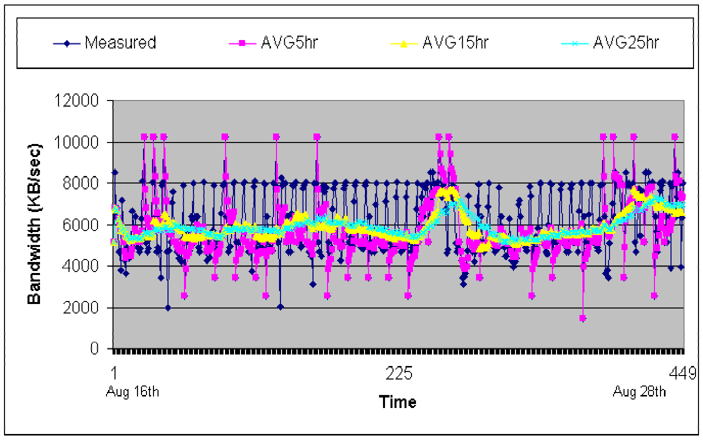

Figure 5: LBL-ANL all transfers and time-based predictors. Depends on request arrival rate.

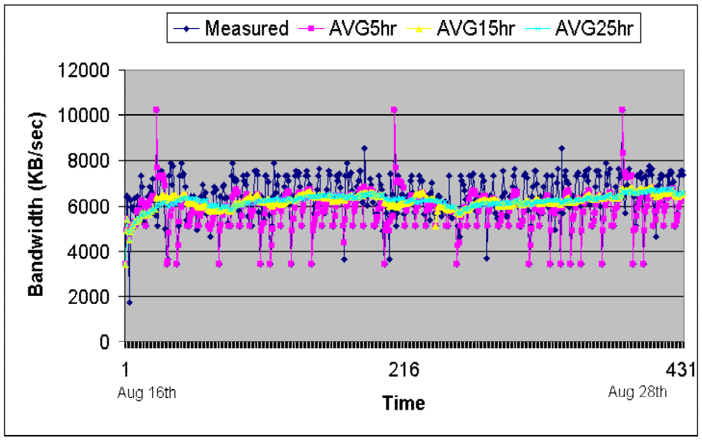

Figure 8: ISI-ANL all transfers and time-based predictors

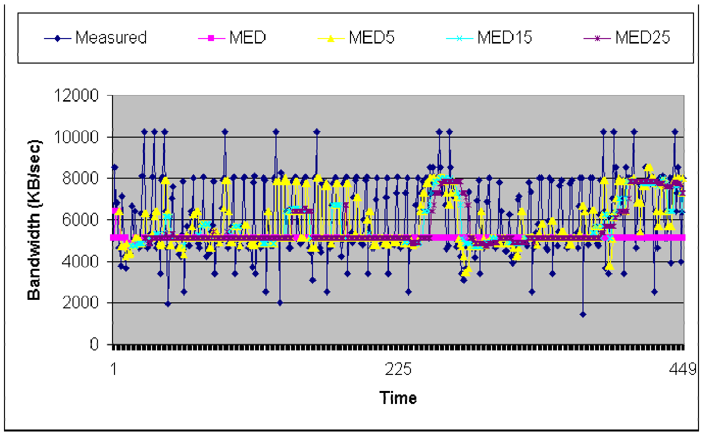

Figure 6: LBL-ANL all transfers and median-based predictors. Similar behavior as in moving averages but averages are more representative of trends.

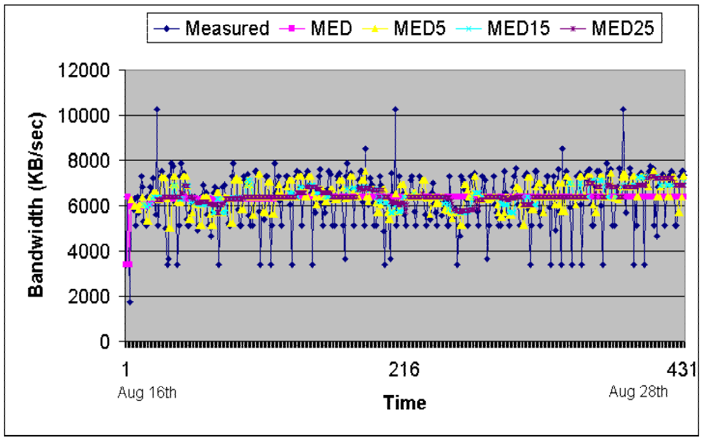

Figure 9: ISI-ANL all transfers and median-based predictors

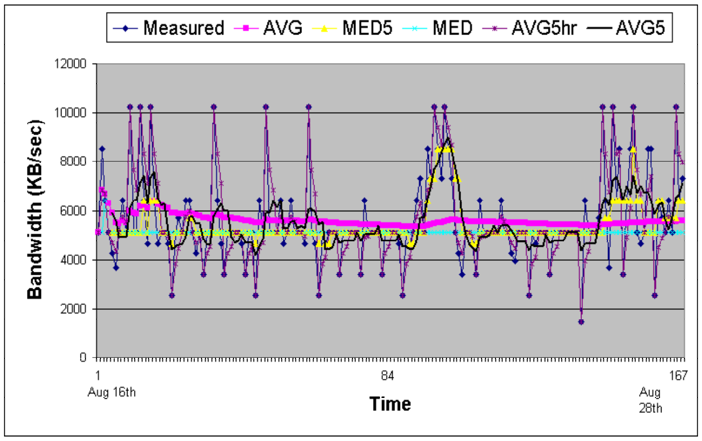

Figure 10: LBL-ANL 10MB transfers

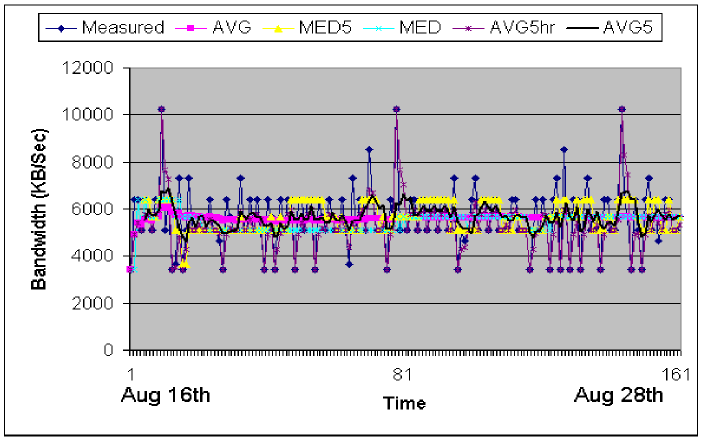

Figure 13: ISI-ANL 10MB transfers

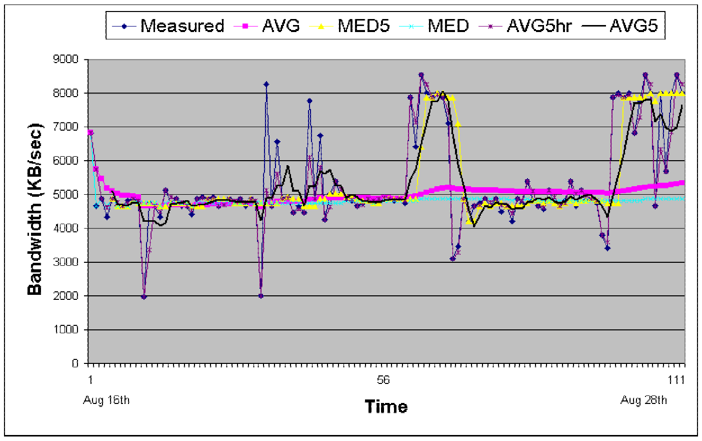

Figure 11: LBL-ANL 100MB transfers

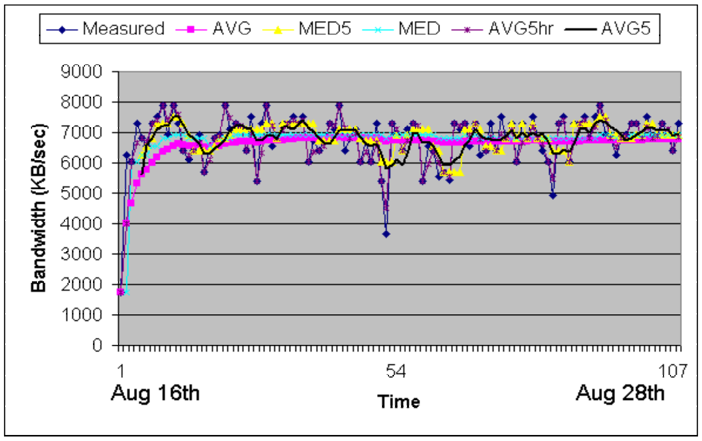

Figure 14: ISI-ANL 100MB transfers

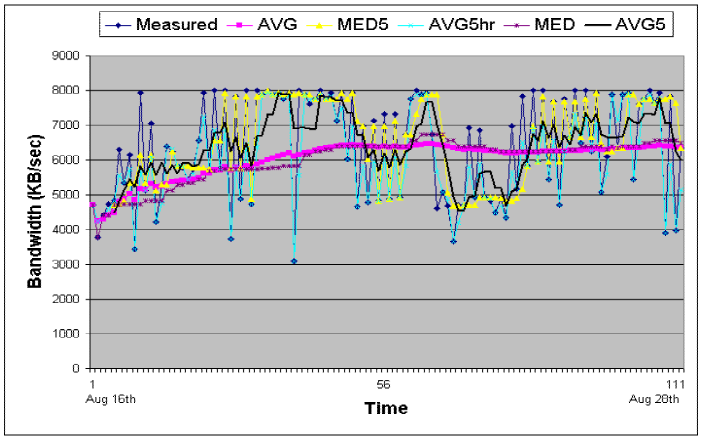

Figure 12: LBL-ANL 500MB transfers

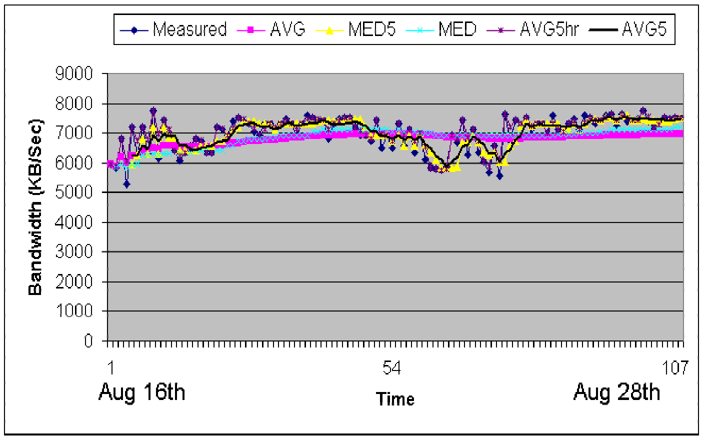

Figure 15: ISI-ANL 500MB transfers

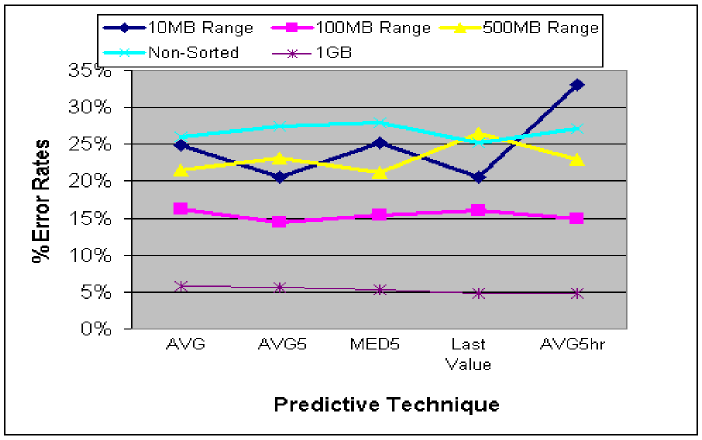

Figure 16: Percent error rates of various predictors for LBL-ANL. Error decreases with classification of files into groups.

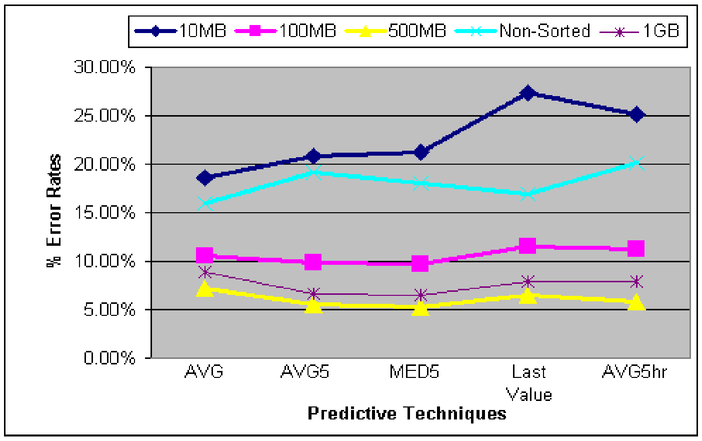

Figure 17: Percent error rates of various predictors for ISI-ANL

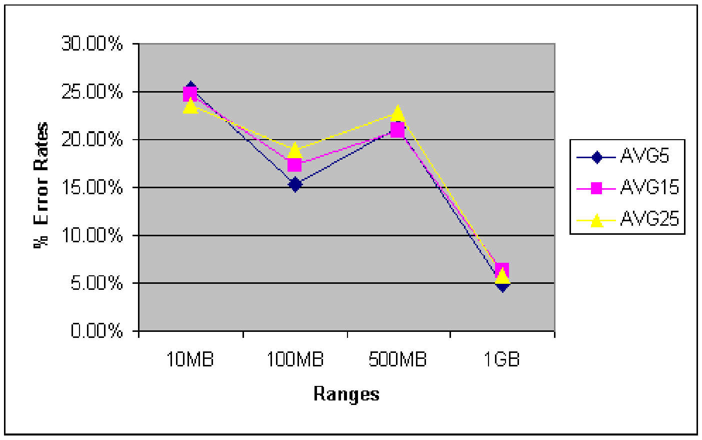

Figure 18: Percent error rates of AVG class predictors for various window sizes for LBL-ANL. Error tends to decrease with smaller optimal windows.