Modeling Remote I/O versus Staging Tradeoff in Multi-Data Center Computing

Ibrahim H. Suslu
Department of Computer Science, North American College, Houston, TX 77038
E-mail: ihsuslu@northamerican.edu

Abstract. In multi-data center computing, data to be processed is not always local to the computation. This is a major challenge especially for data-intensive Cloud computing applications, since large amount of data would need to be either moved the local sites (staging) or accessed remotely over the network (remote I/O). Cloud application developers generally chose between staging and remote I/O intuitively without making any scientific comparison specific to their application data access patterns since there is no generic model available that they can use. In this paper, we propose a generic model for the Cloud application developers which would help them to choose the most appropriate data access mechanism for their specific application workloads. We define the parameters that potentially affect the end-to-end performance of the multi-data center Cloud applications which need to access large datasets over the network. To test and validate our models, we implemented a series of synthetic benchmark applications to simulate the most common data access patterns encountered in Cloud applications. We show that our model provides promising results in different settings with different parameters, such as network bandwidth, server and client capabilities, and data access ratio.

1. Introduction

Data-intensive Cloud applications usually take place in two phases: data generation/collection and data analysis. In the data generation/collection phase, large amounts of data are generated by applications running on different resources or collected from remote instruments, such as sensors, detectors, social networks or other Web applications. In the data analysis phase, collected data is analyzed and another set of data may be generated. If in any of the data collection or analysis phases more than one data centers are involved, efficient data access mechanisms would be an important factor in the end-to-end performance of these applications.

There are mainly four different data access techniques Cloud application developers consider: (i) accessing data streams remotely over the network (remote I/O); (ii) moving application where data is located; (iii) moving data where application runs (staging); and (iv) moving both data and application to an intermediate location (hybrid model). Each data access technique has advantages and disadvantages based on the characteristics of the environment and applications. Therefore, based on application needs, an appropriate data access technique should be used.

In most Cloud applications, the datasets required by the application are either transferred to a temporary space close to the computation (staged-in) or accessed remotely over the network (remote I/O). Due to the nature of applications and the interconnects sites, there are many factors affecting end-to-end performance such as I/O access pattern of the application, size of dataset needed, and network characteristics. Although both data access techniques are widely...
used, the Cloud application developers generally choose one over the other intuitively without making any scientific comparison specific to their application since there is no generic model available that they can use. To the best of our knowledge, there has not been an extensive study comparing both data access techniques and providing clear guidelines of which method to use in which particular case.

In this paper, we propose a generic model for the Cloud application developers which would help them to choose the most appropriate data access mechanism for their specific application workloads. The current approach to find the best data access method is based on active learning. First, the application needs to be run in the same environment with all possible combinations. Once the combination that is the best fit for that environment is discovered, the correct data access technique can be found. Our model, however, provides the best data access technique before running the application.

We initially define the parameters that potentially affect the end-to-end performance of the multi-data center Cloud applications which need to access large datasets over the network. To test and validate our models, we implement a series of synthetic benchmark applications to simulate the most common data access patterns encountered in Cloud applications. We show that our model provides promising results in different settings with different parameters, such as network bandwidth, server and client capabilities, and data access ratio. Cloud application designers and developers can use our model to decide which data access technique is right for their application.

2. The Theoretical Model

Modeling the data access techniques in a Cloud computing environment is not a well-studied area. Staging the data before it is needed can be very complicated because of the dynamic nature of the Cloud environment such as multi-data center configuration, dynamic resource allocation, and the real-time network traffic between sites. Limitations of bandwidth, storage space at certain sites, and data specifications can affect the staging of the data.

In order to model the tradeoff between staging and remote I/O, we first define the parameters that can affect the end-to-end performance of the multi-data center Cloud applications. The application specific values for these parameters should be provided by the user. And developed model would need to evaluate these parameters not only individually, but also as a combination to find out which data access technique is the best for the specific application. The parameters of interest are listed below:

- $R_l$: time to read from local disk to local memory
- $R_r$: time to read from remote disk to remote memory
- $W_l$: time to write from local memory to local disk
- $W_r$: time to write from remote memory to remote disk
- $N_s$: time to send data over network via a staging protocol
- $N_r$: time to send data over network via a remote I/O protocol
- $E$: time spent for computation
- $P$: total time spent for the application

The total time spent for the application via staging would be:

$$P_s = P_{in} + E + P_{out}$$

where

$$P_{in} = R_{rin} + N_{sin} + W_{lin} + R_{lin}$$
and

\[ P_{\text{out}} = Wl_{\text{out}} + Rl_{\text{out}} + Ns_{\text{out}} + Wr_{\text{out}} \]  \hspace{1cm} (3)

On the other hand, the total time spent for the application via remote I/O would be:

\[ P_r = Rr_{\text{in}} + Nr_{\text{in}} + E + Nr_{\text{out}} + Wr_{\text{out}} \]  \hspace{1cm} (4)

The actual computation time for both methods would be the same. The time difference in total (end-to-end) application time comes from network and disk I/O operations. For simplicity, we compare Input and Output operations separately.

In such a setting, for remote I/O to be more efficient than staging, we should have:

\[ Rr_{\text{in}} + Nr_{\text{in}} < Rr_{\text{in}} + Ns_{\text{in}} + Wl_{\text{in}} + Rl_{\text{in}} \]  \hspace{1cm} (5)

and

\[ Nr_{\text{out}} + Wr_{\text{out}} < Wl_{\text{out}} + Rl_{\text{out}} + Ns_{\text{out}} + Wr_{\text{out}} \]  \hspace{1cm} (6)

From Equation 5, we would get:

\[ Nr_{\text{in}} - Ns_{\text{in}} < Wl_{\text{in}} + Rl_{\text{in}} \]  \hspace{1cm} (7)

and similarly from Equation 6, we would get:

\[ Nr_{\text{out}} - Ns_{\text{out}} < Wl_{\text{out}} + Rl_{\text{out}} \]  \hspace{1cm} (8)

which means the time difference coming from using a specialized data transfer protocol versus a remote I/O protocol should be less than the overhead of extra read/write to the disk in staging. In other words, if your remote I/O library performs good in data transfer over network, or your local disk performance is slow, remote I/O might be advantageous over staging. Otherwise, staging method would perform better.

According to authors knowledge, Nrin and Nrout can not be measured separately. Since, these variables are plays important role in our initial mode, we would like to start with two separate models for each data access technique using our measurements with regression analysis in the following section.

3. A Realistic Model based on Regression

In order to define the model for the remote I/O vs staging data access techniques, a standard multiple regression analysis was performed using PASW Statistics 18 [1].

3.1. Regression Model for Data Staging

End-to-end performance of the application (Ps) is a dependent variable whereas Rrin, Nsin, Wlin, Rlin, Wlout, Rlout, Nsout, Wrout, DS (data size), AT (Access Techniques), NA (Network Architectures) were the independent variables for data staging technique model. Full, half, quarter, and eighth are grouped as the data size (DS). NA variable indicates the network architectures such as LAN, CAN, MAN, WAN1, WAN2, WAN3, WAN4, WAN5, WAN6, WAN7. AT variable which is data access techniques (sequential, jump, and random) indicates categorical type of variable. Therefore, (j-1) approach of regression, when including categorical variable in the model, applied [2]. It is also suggested by Kleinbaum [3] et al. when including a nominal independent variable since it can help to index categories of the nominal variable in regression analysis. This method is also called reference cell coding and it is about using dummy variables for one less of the number of categories (j-1). In this study, AT variable which has 3 categories is
recoded into 2 dummy variables (ATj and ATr) that have values of 0 or 1. ATj dummy variable is for jump access technique and ATr dummy variable is for random access technique. The sequential access technique is not coded in a separate variable since it is defined when both ATj and ATr variables equal to 0. The intercept of the regression model will indicate the coefficient for the sequential category.

The initial investigation of the variables is showed that dependent variable Ps and other independent variables are not normally distributed.

Logarithmic transformation was used for variables Ps, Nsin, Wlin, Rlin, Wlout, Rlout, Nsout, and Wout. Before logarithmic transformation, reflect method was applied for Rrin variable. Because of its relatively negative skewness reflect method helped to transform the distribution into the positive direction then logarithm applied. After the transformations all the skewness and kurtosis levels approached to the normal distribution values. After the logarithmic transformation skewness and kurtosis values improved. This was evident for the other variables. PASW Explore procedure was used to check each variables whether or not they improved in terms of normal distribution.

After transformation and removal of the outliers, residuals indicated normal distribution as it can be seen in Figure 1.

Figure 2 indicates that the residuals are normally distributed after the transformation.

Figure 3 shows the normality, linearity and homoscedasticity assumptions. According to
Tabashnick and Fidell [4], if scatterplot of residuals with standardized predicted dependent variable values display the scores scattered around the zero line it indicates that the normality, linearity and homoscedasticity assumptions are met. As values of the residuals scattered along the 0 line the assumptions were assumed to be met. Moreover, Tate [5] stated that moderate violations of the normality assumption may often be ignored with larger sample size as it does not affect the analysis negatively. Kleinbaum et al. also stated if normality assumption is not badly violated, the results reached by regression analysis will generally be reliable and accurate. In addition, Tabshnick and Fidell [4] noted that violations of linearity and homoscedasticity do not invalidate the analysis. Regarding the 595 observations, the sample was large enough and it was concluded that assumptions of regression analysis were met.

By using Mahalanobis distance with \( p < .001 \), multivariate outliers were checked for a number of variables in the model and no multivariate outliers are detected in the data. In fact, maximum Mahalanobis distance were 31.366 and it was less than the \( X^2 \) critical value. \( N = 595 \) and no missing data observed.

The results also indicated multicollinearity (high correlation among independent variables) was an issue. However, some of the independent variables were highly correlated each other: \( N_{soutL} \) and \( N_{sinL} \) (.832) and, \( R_{linL} \) and \( R_{loutL} \) (1.), \( W_{loutL} \) was highly correlated with \( R_{linL} \) (.912) and with \( R_{loutL} \) (.912).

When regression analysis was applied it was observed that \( R_{lin} \) variable is excluded from the analysis by the statistical program due to perfect correlation with the \( R_{lout} \) variable. Since \( R_{lout} \) and \( R_{lin} \) are identical in their relationship with the performance (dependent variable) no further solution was attempted to include \( R_{lin} \) in the model. Moreover, including perfectly correlated independent variables in the model causes greater standard errors and coefficients will be shown as not significant in the model.

The results indicate that an overall model of eleven predictors that significantly predict the performance in staging data technique access, \( R^2 = .984 \) and adjusted \( R^2 = .983 \) \( F(11,583) = 3197.962 \) \( p < .001 \). This model is accounted for 98 % of the variance in performance of staging data access technique. In other words, 98 % of the variance in the PsL was explained by \( R_{rin} \), \( W_{lin} \), \( W_{lout} \), \( N_{sout} \), \( W_{rout} \), \( DS \), \( AT \), and \( NA \).

Since there was a perfect relationship (1.) between the \( R_{linL} \) and \( R_{loutL} \), \( R_{linL} \) was excluded from the analysis by a statistical program due to the multicollinearity. Since the synthetic application creates the same amount of data, \( R_{lout} \) and \( R_{lin} \) operate the same amount of data. It is not surprising that both variables have strong relationship. T test statistics results indicate

![Figure 3. Staging Regression Standardized Predicted Values After the Transformations](image)

---

**Figure 3.** Staging Regression Standardized Predicted Values After the Transformations
that coefficients of the all independent variables are significant except \(W_{\text{outL}}\). However, since multicollinearity is normal with this type of data. It doesn’t affect the inclusion of this coefficient in the regression equation.

\(P_{\text{SL}}\) is estimated by the following regression model.

\[
P_{\text{SL}} = 1.133 + (-.240)(R_{\text{rinL}}) + .268(N_{\text{sinL}}) + .254(W_{\text{inL}}) + (-.009)(W_{\text{outL}}) + (-.157)(R_{\text{outL}}) + (.269)(N_{\text{outL}}) + .290(W_{\text{outL}}) + (.051)(D_S) + .003(N_A) + .054(A_{Tj}) + .154(A_{Tr})
\]

3.2. Regression Model for Remote I/O

Performance of the application \((P_r)\) was the dependent variable whereas \(R_{\text{rin}}, N_r, \) and \(W_{\text{out}}\), \(D_S\)(data size), \(A_T\)(Access Techniques), \(N_A\) (Network Architectures) were the independent variables for remote data access techniques in distributed environments.

\(D_S\) variable indicated transaction and data size including Full, half, quarter, and eighth (DS). \(N_A\) variable includes the network architectures such as LAN, CAN, MAN, WAN1, WAN2, WAN3, WAN4, WAN5, WAN6, and WAN7. \(A_T\) variable which were data access techniques (sequential, jump, and random) indicated categorical type of variable. Therefore (j-1) approach of regression or in other term reference cell coding, when including categorical variable in the model was applied in PASW.

The initial investigation of the variables showed that dependent variable \(P_r\) and other independent variables were not normally distributed. In fact, most of the variables were positively skewed. Logarithmic transformation was used for variables \(P_r, R_{\text{rin}}, N_r, \) and \(W_{\text{out}}\). After the transformations, all the skewness and kurtosis levels approached to the normal distribution values. Also, "L" is added to end of the variable name to indicate log(logarithmic) transformation. In perfectly normal distribution, skewness and kurtosis levels are supposed to be 0. Before the transformations, skewness and kurtosis for the dependent variable \(P_r\) were 2.845 and 7.046 respectively. After the log transformation, skewness and kurtosis values improved and became .360 and -1.270 respectively. PASW Explore procedure was used to check each variable whether or not they improved in terms of normal distribution.

Regression results indicated an overall model of nine predictors that significantly predict the performance in remote I/O data access technique. This model is accounted for nearly 98 % of the variance in performance of remote I/O data access technique. The coefficient of the intercept (constant) captures the coefficient of the sequential access technique as it was not included as a dummy variable in the equation because of the (j-1) dummy variable approach. The coefficient of jump access technique \((A_{Tj})\) is the difference in means between jump and sequential. Similarly, the coefficient of random access technique for \(A_{Tr}\) is the difference in means between random and sequential.
4. Model Validation

Two real-life applications, Hurricane Data Archive and Blast, are used to test the validity of our model. In addition to these real-life applications, we have also used modified versions of our synthetic applications to test some extreme cases. The following sections provide the tests and their results.

4.1. Real-Life Applications

4.1.1. Data Archive for Coastal Science

Coastal/Hurricane research group on CCT has been developing a Simulated Hurricane Database [6] hosted on Petashare[7] containing data produced from ADCIRC [8] application. Initially, archive database is populated by ADCIRC runs for hurricanes and tropical storms that have occurred in the Golf of Mexico over the past 50 years. Then, archive database provides information to additional application for hypothetical storm events. Some applications are available to analyze the simulated hurricane database. We will use the application program to find out which remote data access technique is the best method.

The simulated hurricane database is around 38 GByte. Spider is used as an execution node, and eric is used as a data server node. So, we have used CAN network architecture for this experiment. We have staged in all the data files from eric to spider by globus-url-copy. After the execution, we have staged out all the files from spider to eric back in staging tests. We have repeated the tests five times to eliminate network affects. Application program has accessed the simulated hurricane database which is on eric with parrot/gsiftp remote I/O protocol. This application accesses small portion of the database, so we have used eighth data size. The application reads the database sequentially.

Figure 5 provides both staging and remote I/O results. Remote I/O has better performance than staging. When we replace the variables with their corresponding values, the equation becomes:

\[ Pr = 0.735 + (-0.160)(R_{inL}) + 0.527(N_{rL}) + 0.474(W_{outL}) + (0.065)(D_S) + 0.011(N_A) + 0.080(A_{Tj}) + 0.322(A_{Tr}) \]

When we replace the variables with their corresponding values, the equation becomes:

\[ Ps = 239.126 \quad (9) \]

When we replace the variables with their corresponding values, the equation becomes:

\[ Pr = 92.64 \quad (10) \]

Our model also shows that remote I/O has better performance score. Real-life application confirms the validity of our model.

4.1.2. Blast

Blast compares a sequence with those contained in nucleotide and protein databases by aligning the sequence with previously characterized genes. It finds regions of sequence similarity, which will yield functional and evolutionary clues about the structure and function of this sequence.

Blast is used for the second real-life application. Blast DNA database refseq_rna is used, it is around 1.3 GByte. Spider is used as an execution node and Queen Bee is used as a data server node (MAN). This application accesses all database, so we have used full data size. The application reads the database sequentially.

According to the Figure 6, remote I/O also performing better then staging in our second real-life application.
When we replace the variables with their corresponding values, the equation becomes:

\[ Ps = 8.83 \] (11)

When we replace the variables with their corresponding values, the equation becomes:

\[ Pr = 5.84 \] (12)

Our model also shows that remote I/O has better performance score. Real-life application confirms the validity of our model.

4.2. Synthetic Applications

We have used modified versions of our synthetic applications to test the validity of our models with different use cases. We will report two use cases in this section: full ratio and eighth ratio.

4.2.1. Synthetic Application with Full Ratio

We have used our synthetic application and synthetic data. Spider is used as an execution node and Painter is used as a data server node (WAN4). This application accesses all data, so we have used full data size. The application reads the database randomly.

According to the Figure 7, staging is performing better then remote I/O in our first case synthetic application.

We applied our models as follows:

Staging
When we replace the variables with their corresponding values, the equation becomes:

$$P_s = 12.70$$  \hspace{1cm} (13)

**Remote I/O**

When we replace the variables with their corresponding values, the equation becomes:

$$P_r = 1419.17$$  \hspace{1cm} (14)

Our model shows that staging has better performance score. Synthetic application confirms the validity of our model.

4.2.2. Synthetic Application with Eighth Ratio  We have used our synthetic application and synthetic data. Spider is used as an execution node and Painter is used as a data server node (WAN4). This application accesses eighth of the data, so we have used eighth data size. The application reads the database randomly.

According to the Figure 8, staging is performing better then remote I/O in our second case synthetic application.

We applied our models as follows:

**Staging** When we replace the variables with their corresponding values, the equation becomes:

$$P_s = 5.61$$  \hspace{1cm} (15)
When we replace the variables with their corresponding values, the equation becomes:

\[ Pr = 164.585 \]  \hspace{1cm} (16)

Our model also shows that staging has better performance score. Synthetic application confirms the validity of our model.

4.3. Extreme Cases

We have used modified versions of our synthetic application to test some extreme use cases as well. We will report two extreme use cases in this section. For each case, we run the simulation for two different data access techniques (sequential and random). On the first case, our simulation reads all data and produces 1/100 output data ratio. On the second case, our simulation reads 1/100 input data ratio and produces the same amount of data.

4.3.1. Full Ratio Input, 1/100 Ratio Output

We have used our synthetic application and synthetic data. Spider is used as an execution node and Quinbee is used as a data server node (MAN). This application accesses all data, so we have used full data size.

According to the Figure 9, remote I/O is performing better than staging in our first extreme case synthetic application.

We applied our models as follows:

**Staging** When we replace the variables with their corresponding values, the equation becomes:

\[ Ps = 6.50 \]  \hspace{1cm} (17)

**Remote I/O**

When we replace the variables with their corresponding values, the equation becomes:

\[ Pr = 5.45 \]  \hspace{1cm} (18)

Our model shows that remote I/O has better performance score. Synthetic application confirms the validity of our model.

According to the Figure 10, staging is performing better than remote I/O in our first extreme case synthetic application with random data access.

We applied our models as follows:

**Staging** When we replace the variables with their corresponding values, the equation becomes:

\[ Ps = 5.78 \]  \hspace{1cm} (19)
Figure 10. Extreme Case Synthetic Application Results with Full Input 1/100 Output Random

Figure 11. Extreme Case Synthetic Application Results with 1/100 Input 1/100 Output Sequential

Remote I/O When we replace the variables with their corresponding values, the equation becomes:

\[ Pr = 1296.52 \]  \hspace{1cm} (20)

Our model shows that staging has better performance score. Synthetic application confirms the validity of our model.

4.3.2. 1/100 Ratio Input, 1/100 Ratio Output We have used our synthetic application and synthetic data. Spider is used as an execution node and Quinbee is used as a data server node (MAN). This application accesses 1/100 data, so we have used eighth data size.

According to the Figure 11, remote I/O is performing better then staging.

We applied our models as follows:

Staging
When we replace the variables with their corresponding values, the equation becomes:

\[ Ps = 3.50 \]  \hspace{1cm} (21)

Remote I/O
When we replace the variables with their corresponding values, the equation becomes:

\[ Pr = 2.51 \]  \hspace{1cm} (22)
Our model shows that remote I/O has better performance score. Synthetic application confirms the validity of our model.

According to the Figure 10, staging is performing better then remote I/O with random data access.

We applied our models as follows:

Staging
When we replace the variables with their corresponding values, the equation becomes:

\[ Ps = 4.13 \] (23)

Remote I/O
When we replace the variables with their corresponding values, the equation becomes:

\[ Pr = 162.50 \] (24)

Our model shows that staging has better performance score. Synthetic application confirms the validity of our model.

5. Related Work
According to Stockinger [9], the entire resource selection problem requires detailed cost models with respect to data transfer. A cost model for data-intensive applications is discussed in [10] where theoretical models for data intensive job scheduling are presented. The metric for measuring efficiency is the effective time seen by the client application. The model includes all important factors in a distributed Data Grid and takes various storage and access latency into account to determine optimal data access.

More general performance engineering approaches are discussed in [11]. In this work, they analyze a typical distributed system and point out performance analysis aspects in order to improve the overall job execution time of the system. They give a detailed look at the following two domains: data access and networking.

Data staging and remote I/O have been compared by different studies from different aspects. For instance, GridFTP as a staging technique and RFIO as a remote I/O technique have been compared by Kalmady and Tierney [12] on wide area networks. According to Kalmady and Tierney, the performance of RFIO is better than gsiftp for one stream. Gsiftp performs better on multiple streams. With tuning, RFIO becomes pretty close to gsiftp which means that proper tuning can make the difference. They came up with the following observations: i) setting the TCP buffer size a proper value is the most important factor for a good performance; ii) 2-3 parallel streams will gain an additional 25% performance over a single tuned stream; iii) a mechanism of dynamically varying the buffer sizes during data transfer is needed, because of sensitivity of the variation in network traffic; iv) the same throughput can be gained with tuned buffers using untuned TCP buffers with enough parallel streams.

Thain and Livny compared performance of parrot/chirp with other staging and remote I/O techniques on the study [13] with Andrew-like benchmark. According to the authors, separating computation from storage makes I/O cost high. Copying data gets slower over the network, but the slowdown in the network makes staging acceptable because of increased throughput via remote parallelization. Also, authors point out that the differences in the performance between Chirp, ftp, and NeST are because of the cost of the metadata lookups.

Modeling the data access techniques in a distributed environment is not a well-studied area. Staging the data before it is needed can be very complicated because of the dynamic nature of the distributed environment such as real-time network traffic and congestion. Limitations of bandwidth, storage space at certain sites, and data specifications can affect the staging of the data. A mathematical model for a basic data staging problem is studied by Theys et al.
All parameter values for the network and data request a stay fix in the scheduling process. However, the parameter values for the model are changed temporarily to reflect the dynamic nature of the distributed environment. The parameters can be categorized by the node storage capacity and node number, the link availability starting and ending time, bandwidth, latency, source node, and destination node. Also, every request has data size, list of sources, and list of destinations. Elwasif et al. [15] developed a model for farming applications with and without server side staging to analyze the effect of staging, and verify the model with experiments and simulations.

A remote I/O performance model [16] can estimate remote I/O cost before performing the application, so the application can be evaluated better. The paper also presented the design of a remote I/O performance predictor that gives the user a concept how much remote I/O costs for the application, so the appropriate parameters can be chosen for the application. A practical remote data access model is presented in [17] which describes a tightly coupled, a data oriented infrastructure approach with building a leading edge technology to provide very high-speed, and widespread access to large data storage. Antoniu et al. [18] provide a transparent data access model which enables users to access data via global identifiers. This model manages data persistence dynamically and transparently in distributed environment using two approaches which are distributed shared memory and peer to peer systems.

Shivam et al. [19] present the Non-Invasive Modeling for Optimization (NIMO) system which automatically learns cost models for predicting the execution time of computational science applications on distributed environments. NIMO first generates training samples for distributed applications, then by using these samples learns cost models with statistical learning techniques. NIMO is an active system, so it deploys and monitors the sampled application under different conditions. NIMO is also non-invasive so it collects training data from passive streams without effecting not only the operating system or the application, but also the application source, or library.

On the other hand, there are some challenges that arise based on NIMO. According to Shivam et al. [19], sampling acquisition may have high overhead. Also, the number of samples needed, given the level of accuracy, increases exponentially. The training sample set may not represent the entire operating range of the system.

Although our approach and NIMO can suggest which data access method to use, there are significant differences between them. Researchers can use our approach before the design phase, so they can design the application based on the best data access technique. NIMO does not only evaluate the data access methods, but also the active system elements such as CPU, cache, and I/O system behaviors. Sometimes, NIMO can prefer a data access technique which is worse than others in terms of overall throughput, but preferable in terms of other active elements such as CPU, cache, and I/O.

6. Conclusions and Future Work
We have developed generic models and set guidelines for the application developers which would help them to choose the most appropriate data access method for their application. We defined the parameters that potentially affect the end-to-end performance of the distributed applications which need to access remote data.

Sequential and jump data access always performs better than random data access on all network architectures. High-speed networks improve the data transfer ratio dramatically. Random data access on remote I/O is not benefiting from high-speed network architectures, so the performance decreases dramatically. The differences on sequential and jump is less than the differences on random. So, researcher should be more careful when remote I/O was chosen. Since all data should be staged in before the execution, decreasing the data ratio improves the remote I/O performance.
Increasing the distance between execution node and data server node with high-speed network decrease the the gap between sequential and jump performance. Without high-speed network, it decreases the gap between remote I/O and staging on all data access techniques. On the other hand, it increases the gap on smaller data ratios. Also, choosing proper remote I/O protocol is a crucial decision for the end-to-end application performance.

References

[1] 2010 www.spss.com URL www.spss.com
[2] Norusis M J 2008 SPSS 16.0 Guide to Data Analysis (prentice hall Inc.)
[3] Kleinbaum D G 1998 Applied Regression Analysis and Other Multivariable Methods (Duxbury Press)
[4] G T B and S F L 2007 Using multivariate statistics (Pearson Education Inc.)
[5] Tite R 1996 General linear model applications
[6] Bhagawaty R S H, Jiang L, Dubbaka S, Hu K, Pothanis S, Allen G, Brener N, Chen Q and Kosar T 2010 CLADE
[7] 2010 http://www.petashare.org/ URL http://www.petashare.org/
[8] http://adcirc.org/ URL http://adcirc.org/
[9] Stockinger H May 2005 Data management in data grids - habilitation overview Tech. rep. Research Lab for Computational Technologies and Applications
[10] Stockinger H, Stockinger K, Schikuta E and Willers I February, 2001 9th Euromicro Workshop on Parallel and Distributed Processing (PDP 2001), IEEE Computer Society Press, Mantova, Italy
[11] Laure E, Stockinger H and Stockinger K 2005 Journal of Concurrency and Computation: Practice and Experience, Wiley Press 17(2-4)
[12] Kalmady R and Tierney B 2001 Proceedings of Computers in High Energy Physics
[13] Thain D and Livny M 2003 Univ. of Notre Dame, Computer Science and Engineering Dept.
[14] Theys M D, Siegel H J, Beck N B, Ta M and Jurczyk M 1998 Proceedings of the Seventh Heterogeneous Computing Workshop
[15] Elwasif W, Plank J and Wolski R 2001 Proceedings of the 1st International Symposium on Cluster Computing and the Grid
[16] Shen X, keng Liao W and Choudhary A 2001 IASTED Applied Informatics, Innsbruck, Austria
[17] Andrews P M, Sherwin T and Banister B 2003 Proceedings on 20th IEEE/11th NASA Goddard Conference
[18] Antoniu L G and BougB M J 2005 Scalable Computing: Practice and Experience
[19] Babu S, P Shivam and Chase J 2006 International Conference on Very Large Data Bases (VLDB)