A Novel Workload Migration Scheme for Heterogeneous Distributed Computing

Yawei Li and Zhiling Lan
Department of Computer Science
Illinois Institute of Technology, Chicago, IL 60616
{liyawei, lan}@iit.edu

Abstract

Dynamically partitioning of adaptive applications and migration of excess workload from overloaded processors to underloaded processors during execution are critical techniques needed for distributed computing. Distributed systems differ from traditional parallel systems in that they consist of heterogeneous resources connected with shared networks, thereby preventing existing schemes from benefiting large-scale applications. In particular, the cost entailed by workload migration is significant when the excess workload is transferred across heterogeneous distributed platforms. This paper introduces a novel distributed data migration scheme for large-scale adaptive applications. The major contributions of the paper include: (1) a novel hierarchical data migration scheme is proposed by considering the heterogeneous and dynamic features of distributed computing environments; and (2) a linear programming algorithm is presented to effectively reduce the overhead entailed in migrating excess workload across heterogeneous distributed platforms. Experiment results show that the proposed migration scheme outperforms common-used schemes with respect to reducing the communication cost and the application execution time.

1. Introduction

The emerging distributed computing environments, such as the Computational Grids [9,13,20], have been rapidly becoming the dominant computing platforms for large-scale simulations. Many large-scale applications are adaptive in that their computational load varies throughout the execution and causes uneven distribution of the workload at run-time. Efficiently *partitioning* of the application data and *migration* of excess workload from overloaded processors to underloaded processors during execution are critical techniques needed for efficient use of distributed computing environments. In some literature, these are also called dynamic load balancing (DLB) techniques. Tradition research has been focused on *partitioning* problem; however, the cost entailed by workload (or data) *migration* may consume orders of magnitude more time than the actual partitioning when the excess workload is transferred across geographically distributed machines. In particular, with workload migration, it is critical to take into account that the wide area network (used to connect the geographically distributed sites) performance is dynamic, changing throughout execution, in addition to considering the resource heterogeneity. In this paper, we present a novel workload migration scheme aiming at reducing data transferring time across heterogeneous distributed platforms for distributed computing. Transferring excess workload in a distributed environment is complicated due to the following challenges imposed by the nature of distributed systems:

- Data migration in distributed environments inevitably involves communication within a cluster and across clusters, where the network performance gap could be orders of magnitude different due to the two-layered interconnects architecture [18]. Neglecting this difference, as done in traditional application-level load balancing schemes, significant application performance degradation is entailed due to the overhead of inter-cluster links.
- The shared networking resources are highly dynamic and unstable, which makes it difficult to make an accurate data transferring decision responsive to resource fluctuation.

Our proposed migration scheme called DistPM combines a novel hierarchical approach with real-time performance evaluation. to deal with the heterogeneous and dynamic features of distributed systems. In particular, a heuristic method based on a linear programming algorithm is developed to reduce the overhead entailed in

migrating workload over shared networks across heterogeneous distributed platforms. The proposed migration scheme is evaluated with both synthetic datasets and real MPI applications on the nation-wide production system TeraGrid [20]. Our experiments show that DistPM outperforms common-used schemes with respect to reducing the communication cost and the application execution time.

The rest of the paper is organized as follows. Section 2 discusses related works. Section 3 presents a. problem formulation for expressing application partitioning across heterogeneous distributed platforms. Section 4 describes our proposed workload migration scheme for distributed computing. In section 5, experiments with both synthetic datasets and real applications on the nation-wide TeraGrid are presented. Finally, we conclude our investigation in section 6.

2. Related Works

Research works in performance-oriented distributed computing has been focused on system-level load balancing or task scheduling, [2,14,24]. They aim to maximizing the overall throughput or average response time of the systems. Fewer works have been done with the context of application-level load balancing because it is difficult to model application-level performance in a heterogeneous and dynamic environment.

Furthermore, most application-level load balancing schemes have been focused on application partitioning via graph algorithms. For instance, ParMETIS [19] and JOSTLE [22] are two widely used data partitioning tools. DRUM [8] extends graph partitioner with a resource-aware performance model to optimize both the quality of partition and communication cost by using a heterogeneous scalar node power value. However, it does not address the issue of reducing migration cost, that is, the cost entailed by load redistribution, which can consume order of magnitude more time than the actual computation of a new decomposition. MinEX [6] proposes a latency-tolerant algorithm that takes advantage of overlapping the computation of internal data and the communication of incoming data to reduce data migration cost. Unfortunately, it requires applications to provide such a parallelism between data processing and migration, which restricts its applicability.

S. Genaud [10] enhances the MPI_Scatterv primitive to support master-slave load balancing by taking into consideration the optimization of computation and data distribution using a linear programming algorithm. However, this work is limited to static load balancing.

Y.F. Hu, R.J. Blake and D.R.Emerson [12] propose an optimal data migration algorithm in diffusive dynamic load balancing through the calculation of Lagrange multiplier of the Euclidean form of transferred weight. This distinguished work can effectively minimize the data movement in homogenous environments, plus it does not consider the network heterogeneity.

3. Problem Formulation

In the rest of the paper, the following nomenclature is used:

- V - The set of processors.
- E – The set of communication channels.
- $w_{i \in v}$ - The amount of workload in the processor i before load balancing.
- $w'_{i \in V}$ - The amount of workload in the processor i after load balancing.
- $T(w_i, i)$ - The computation time for processor i to process the workload w_i.
- $T_{comm}(x, i, j)$ - The communication time from processor i to transfer *the* workload x to processor j. DistPM employs the α-β linear model to calculate the communication time incurred by data migration: $\mathrm{Tcomm}(x, i, j) = \alpha_{i,j} + \beta_{i,j} * x$ where $\alpha_{i,j}$ and $\beta_{i,j}$ represent the prediction value of end-to-end communication latency and the transfer rate of the communication channel connecting two processors collected through NWS [22] toolkit.
- $T_{overhead}$ - The cost to perform the data migration operation, including the data redistribution time, synchronization time and data structure rebuilding time. In particular, the data redistribution time is a significant part of this cost, which will be discussed in this paper.
- T - The application execution time without data partition and migration. i.e. $T = max(T(w_i, i))$.
- T' - The application execution time with data partition and migration, i.e. $T = max(T(w'_i, i))$.

The following problem formulation is proposed to summarize the objective of data partition and migration across heterogeneous distributed platforms:

$$\left.\begin{array}{l} \textit{find a workload redistribution such that} \\ \textit{maximize}(T - T' - T_{overhead}) \\ \textit{subject to}: \\ \sum_{i \in V} w_i' = \sum_{i \in V} w_i \\ T' + T_{overhead} < T \end{array}\right\} \quad (1)$$

Here, w_i and w'_i represent the amount of workload before and after load redistribution, T and T' denotes corresponding execution time. The problem can be further transformed to minimize both T' and $T_{overhead}$. Minimizing T' corresponds to finding an optimal workload distribution such that the resulting execution time is

minimized, which is denoted as *data partition*. Minimizing $T_{overhead}$ corresponds to minimizing the cost incurred by transferring excess workload, which is denoted as *load migration*.

In our earlier work, we have proposed a hierarchical partitioning approach for SAMR (Structured Adaptive Mesh Refinement) applications running in heterogeneous distributed environments [15,16,17]. This paper is focused on how to minimize $T_{overhead}$, in particular, the communication cost for transferring excess workload from overloaded processors to underloaded processors.

4. Proposed Migration Scheme

4.1 Overall Design

Typical distributed systems may consist of multiple machines connected by inter-machine networks where the communication cost is up to three or four orders of magnitude more than that in intra-connected networks. Without considering this performance gap between intra-connected and inter-connected networks, a large number of messages and a large volume of data will be incurred across distributed machines. Therefore, the proposed work utilizes the hierarchical approach used in our earlier work in which global operation is performed across clusters and local operation is conducted within clusters.

As shown in figure 2, the *group* concept is used to support hierarchical operation. A group is defined as a set of homogeneous processors connected with dedicated system area networks; a group can be a shared-memory parallel computer, a distributed-memory parallel computer, or a cluster of Linux PCs. Processors involved in distributed computing will be divided into groups as determined by the configuration.

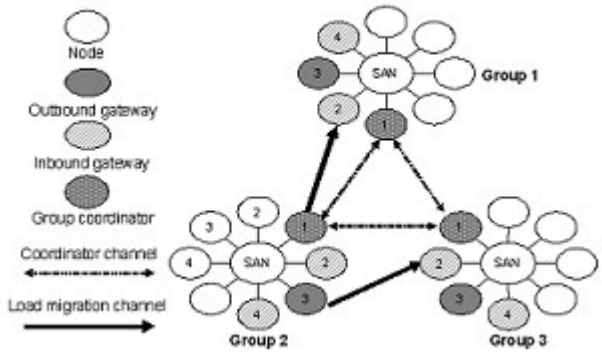

Figure 2. Hierarchical Approach

A salient feature of the proposed work is that the group construction is totally transparent to applications: The proposed work employs the topology-discovery interface of MPICH-G2 to detect the depth communication channel, which represent the communication capability level for each link. Processes sharing the intra-connected links are clustered into a group while processes connected by inter-connected links are separated into different groups. Each group has two special processes: *group coordinators* and *migration gateways*.

- **Group Coordinators**. Each group elects a process as the group coordinator, and a global coordinator is elected from the group coordinators. A coordinator is in charge of collecting group load information, monitoring and predicting both system and application status regarding this group. During each load balancing step, each group coordinator gathers its current load information of the group and reports the information to the global coordinator which then makes a data partition decision with the global knowledge. The principle behind the two-layer coordinators is to maintain the advantages of global-knowledge and scalability in the same time while reducing the number of messages among remote processes.
- **Migration gateways**. In each group, multiple processes are designated as the migration gateways. They are responsible for moving excess workload from overloaded groups to underloaded groups along migration channel. According to the direction of data flow, these gateways can be classified into outbound and inbound ones. For example, in Figure 2, there are one overloaded group 2 and two underloaded group 1 and 3. During data migration step, each gateway combine multiple intra-cluster messages into one single message and then sends it to a corresponding gateway in a remote group. The goal here is to minimize the occurrence of remote communications across clusters with high latency and fully utilize the high bandwidth provided in modern networks.

4.2 Linear Programming Algorithm

Based on the results obtained from data partition phase, groups are divided into three categories: (1) the overloaded set O; (2) the underloaded set U; and (3) the balanced set B. To reduce the overhead, only groups in the set O and U are involved in the data migration phase.

As shown in Figure 2, performance bottleneck of the data migration lies in the inter-group communication. Therefore efficiently planning the amount of migration data along each channel between groups with the goal to minimize $T_{overhead}$ is the major concern here. Because of the advantages of multi-gateway model, which will be discussed in section 4.3, there are total $|O| \times |U|$ inter-group

communication channels between the set of O and U. Let δ_{ij} be the amount of workload transferred from the overloaded unit i to the underloaded unit j. Figure 3 illustrates an example for load migration where there are two overloaded groups and three under-loaded ones. In order to minimize $T_{overhead}$, we need to determine the optimal distribution of δ_{ij}.

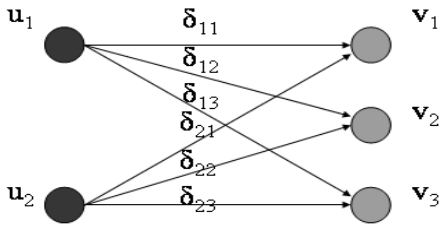

Figure 3. Data migration planning

We propose a heuristic linear programming (LP) algorithm to solve this problem as shown in Equation (2). The idea here is to minimize the longest migrating time among all the channels since all the communications are conducted in parallel. According to the conservation law, for each group i, the summation of workload on all outgoing/incoming channels should be equal to $|w_i - w'_i|$. And the network performance parameter $\alpha_{i,j}$ and $\beta_{i,j}$ are prediction value forecasted using sliding-window mean-based method on historical measurements.

Thus the min-max optimization problem in Equation (2) can be further transformed to a more concise format as shown in Equation (3) by introducing a new variable T_{comm} denoting the maximal data migration time among all the inter-group channels.

$$\left.\begin{array}{l} Minimize(max(\mathrm{T_{comm}}\ (\delta_{\mathrm{i,j}},\mathrm{i,j})\ )), i \in O, j \in U \\ Tcomm\ (\delta_{\mathrm{i},j},\ \mathrm{i},\ \mathrm{j}) = \alpha_{i,j} + \beta_{i,j} * \delta_{i,j} \\ \sum_{i=1}^{|O|} \delta_{ij} = w'_j - w_j \\ \sum_{j=1}^{|U|} \delta_{ij} = w_i - w'_i \\ \delta_{\mathrm{ij}} \geq 0 \end{array}\right\} \quad (2)$$

$$\left.\begin{array}{l} Minimize(T'comm) \\ \forall i \in O, \forall j \in U, T'_{comm} \geq T_{comm}\ (\delta_{i,j},\ i,\ j) \\ Tcomm\ (\delta_{i,j},\ i,\ j) = \alpha_{i,j} + \beta_{i,j} * \delta_{i,j} \\ \sum_{i=1}^{|O|} \delta_{ij} = w'_j - w_j \\ \sum_{j=1}^{|U|} \delta_{ij} = w_i - w'_i \\ \delta_{ij} \geq 0 \end{array}\right\} \quad (3)$$

4.3 Analysis of Multiple Gateways

As the rapid development of network hardware, e.g. the optical network and switch, the speed of network bandwidth has been increasing dramatically while the network latency does not improve significantly [3]. Given this fact, we utilize multiple gateways in our proposed migration scheme with the objective to reduce inter-cluster latency and take advantage of increased bandwidth available in distributed systems.

The advantage of introducing multiple gateways instead of only one is to obtain the apparent speedup from multi-port communication model in the group layer. The performance gain comes from the improvement of communication parallelism, which enables simultaneous workload transferring among multiple groups. In single gateway or proxy model [5], multiple inter-group message transfers are actually serialized due to the typical single-port hardware in one machine even considering the non-blocking and multithreaded communication model [4], which deteriorates the communication performance when considering each inter-cluster communication could be very costly.

Furthermore, recent works [24] indicates the multi-port model is more efficient for inter-connecting links. Therefore the multiple gateways model is adopted to simulate multi-port model in the group layer in order to enable simultaneous data migration between different group pairs allowing multiple parallel TCP stream to make full use of the available backbone bandwidth.

We did an analytical study comparing the non-gateway and multi-gateway model. Due to the space limitation, we only present out the results based on our analysis: unless the amount of intra-cluster communication is huge, e.g. tens of GBytes, in most scenarios it is advantageous to use multiple gateways per group with each handling inbound/outgoing communication with a remote group.

5. Experimental Results

In this section, we evaluate the proposed migration scheme DistPM with both a simulation code with

synthetic dataset and a real application in cosmology [5] on the nation-wide TeraGrid. The experimental data is divided into two categories: reduction of data migration time and reduction of overall execution time.

The synthetic code simulates dynamic workload changes on each processor during each computation iteration, and then invokes data migration after data partition to achieve load balancing.

The real application that we tested is the cosmology application ENZO [7]. ENZO is one of the successful, parallel implementations of the SAMR (Structured Adaptive Mesh Refinement) algorithm. SAMR, developed by Marsha Berger et al. in the early 1980's, is a class of adaptive strategies that address this problem by performing high spatial resolution only in those required regions. ENZO entails the detailed computations that simulate the formation and evolution of cosmic structures such as galaxies and clusters of galaxies from shortly after the big bang to the present day. Such modeling offers the only practical means to test theory against observations and to rule out incorrect hypotheses. ENZO includes solving the coupled equations of gas dynamics, collisionless dark matter dynamics, self-gravity, and cosmic expansion in three dimensions and at high spatial resolution.

The current production mode of the TeraGrid consists of four clusters at CACR, NCSA, SDSC and UC/ANL sites, each of which are composed of IA-64 computation nodes with two CPU with processing rate ranging from 1.3 to 1.5 HZ, and memory size varying from 0.5 GB to 6 GB. The cluster nodes are intra-connected by Myrinet network and the inter-connected network are 40-GB links. And the infrastructure software packages are SuSE Linux, Globus, NWS and MPICH-G2. Figure 4 gives the network architecture of Teragrid. Based on experimental data obtained from MPI benchmark running in the four sites, we calculate the performance of networks within a cluster and across clusters, which is summarized in Table 1. As shown in the table, the performance gap between two-layer networks is obvious.

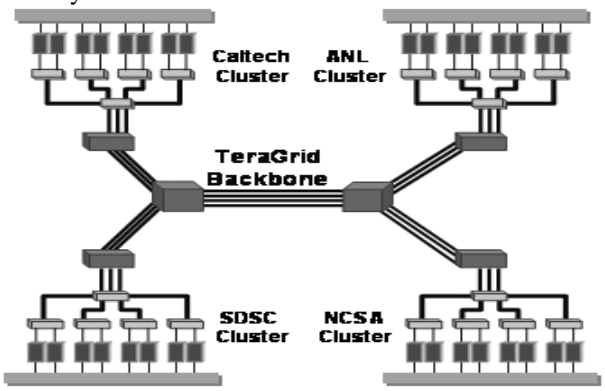

Figure 4. Network architecture of Teragrid

Table 1. Network Performance of Teragrid

	Intra-connected network	Inter-connected network
Latency	2E-3 – 8E-3 ms	3 – 90 ms
Bandwidth	240 – 1600 Mbps	4 – 700 Mbps

5.1 Reduction of Migration Cost

As discussed in section 4, the objective of data migration scheme is to effectively reduce the inter-group communication cost. We ran our simulation code with different migration load sizes on the TeraGrid, with the number of processors ranging from 2 to 64 nodes evenly distributed at NCSA, Caltech, UC/ANL and SDSC sites. We compared our proposed DistPM with the widely-used greedy algorithm [4] that orders the communication channels by capacity and feeds the fastest channel with as much load as possible. Figure 5 shows the advantage of communication cost reduction of our DistPM over the greedy algorithm. Here, the relative improvement is defined as:

$$\frac{\textit{Reduction of Toverhead in Greedy} - \textit{Reduction of Toverhead in LP}}{\textit{Reduction of Toverhead in Greedy}}$$

As shown in Figure 5, our migration scheme has 30%-55% performance gain over the widely used greedy scheme. The greedy algorithm only focuses on partial optimization that may reduce the migration time along some channels at the cost of increasing others, which is exactly LP-based optimization can avoid. Furthermore, it is observed that performance advantage of the proposed DistPM decreases when the migration data size becomes larger. This is due to two major reasons: 1) When messages are small, the latency is more dominant and our scheme can effectively reduce number of inter-group messages so as to reduce the communication cost; 2) The NWS toolkit produces a larger estimation deviation when collecting network performance due to its implementation limitation [21].

We also evaluated our proposed scheme with the ENZO application under the same distributed environments. Here, we compare the communication cost entailed by our proposed DistPM with that entailed by the original ENZO which does not take into consideration the heterogeneous and dynamic features of distributed systems. As shown in Figure 6, the performance gain increases steadily as the number of nodes increases. We also notice that when the number of processors is larger than 24, the performance gain decreases slightly, According to the Amdahl's law [1], the advantage of parallelism reaches its limit because of the application's problem size.

Besides, we compare our multi-gateway model with the single gateway model [5]. Figure 7 illustrates that

multi-gateway achieves significant communication gain over the single gateway model due to the parallelization of communication used in multi-port model.

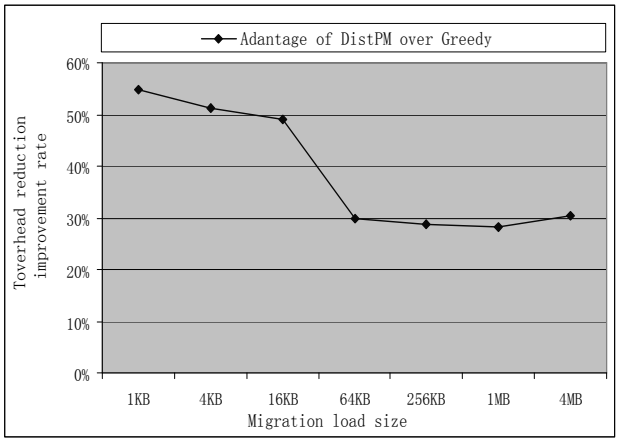

Figure 5. Reduction of Communication Cost with Synthetic Datasets (DistPM vs. Greedy Algorithm).

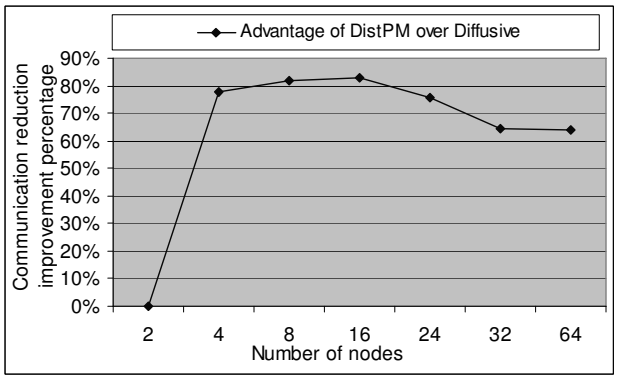

Figure 6. Migration time improvement for ENZO code using Shockpool3d dataset

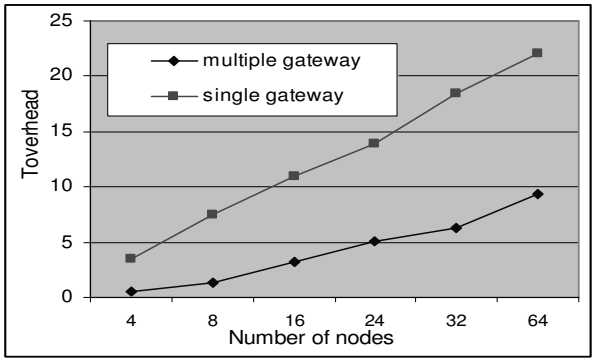

Figure 7. Comparison of $T_{overhead}$ between Multiple-gateway and single gateway.

5.2 Reduction of Execution Time

The second set of experiments is to evaluate whether the proposed migration scheme can effectively minimize the overall execution time of an application.

Figure 8 compares the relative reduction of overall execution times (aka makespan) of the synthetic simulation code using the DistPM optimization and parallel diffusive migration optimization, which does not consider the heterogeneous and dynamic features of distributed computing environments. Here we defined the relative reduction of execution time as:

$$\frac{\textit{Makespan without optimization} - \textit{Makespan using optimization}}{\textit{Makespan without optmiation}}$$

Figure 8 shows our proposed scheme always maintain a stable execution time reduction rate about 20% and good scalability as the size of system increases while the performance of parallel diffusive scheme degrades sharply in large-scale application deployment. This is because the hierarchy structure of our scheme prevents the group number increasing fast so as to suppress the number of message exchanges and load migrations across group, which is quite contrary in diffusive migration scheme.

We also evaluate the performance of the ENZO applications with the proposed migration schemes under different inter-connecting network conditions listed in table 2. As shown in the figure 9, the proposed migration scheme performs much better when network performance is not satisfying, e.g. the latency is high and bandwidth is relative low. This can be explained as in case of low network performance, the communication cost is a more significant part of the overall execution time than that in case of high network performance, and thus the communication optimization of the proposed work contributes much more to the performance gain.

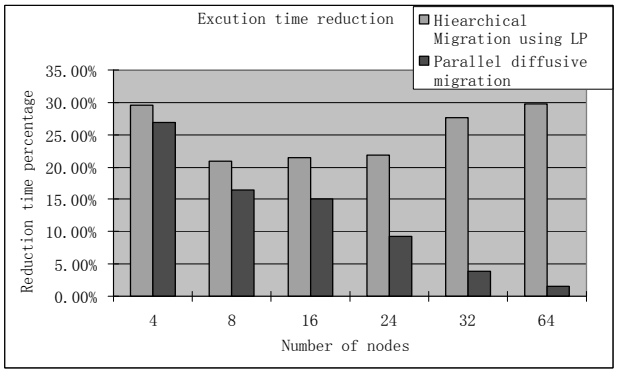

Figure 8. Reduction of Execution Time with Synthetic Datasets (DistPM vs. Diffusive Algorithm)

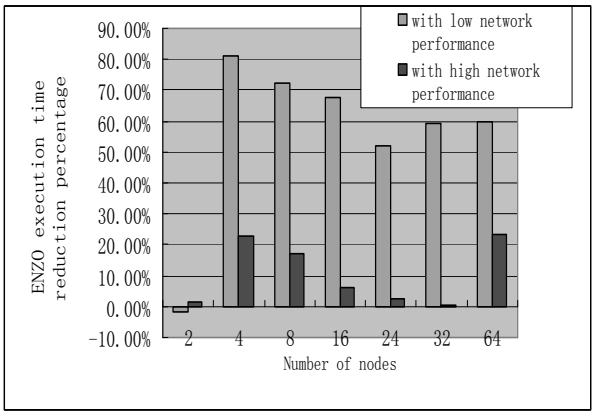

Figure 9. Comparison of ENZO Execution Time Reduction under Different Networking Performance

Table 2. Different inter-connecting network performance conditions in Teragrid

	Latency	Bandwidth
High	3-8 ms	50 – 100 Mbps
Low	45-90 ms	4 – 6 Mbps

6. Conclusions

In this paper, we have proposed a data migration schemes for large-scale adaptive applications running in distributed environments. The linear programming based heuristic algorithm provides the optimized allocation of migrated data along different inter-cluster channels so as to reduce the migration overhead. The hierarchical migration approach further hides inter-cluster latency by using multi-gateway model.

Our future work includes fully evaluating and refining the proposed techniques. For example, to obtain an accurate communication prediction, the multivariable method will be explored. Furthermore, we will evaluate our proposed DistPM with more applications including the molecular dynamics application GROMACS [11].

Acknowledgement

This work is supported by a grant from the National Computational Science Alliance with the NSF PACI program. It is also supported by a grant from NSF NGS 0406328. We would like to acknowledge NCSA, SDSC, Caltech, and UC/ANL for the use of the TeraGrid Clusters.

References

[1] Amdahl, G.M., "Validity of the single-processor approach to achieving large scale computing capabilities," Proceedings of AFIPS Conference, 1967, pp. 483-485

[2] C. Banino, O. Beaumont, L. Carter, "Scheduling Strategies for Master-Slave Tasking on Heterogeneous Processor Platforms", IEEE Transactions on Parallel and Distributed Systems, Vol. 15, No. 4, April 2004

[3] F. Berman, G. Fox and T. Hey, Grid Computing – Making the Global Infrastructure a Reality. 2002 John Wiley & Sons, Ltd ISBN: 0-470-85319-0

[4] Prashanth B. Bhat , Viktor K. Prasanna , C. S. Raghavendra, "Adaptive communication algorithms for distributed heterogeneous systems", Journal of Parallel and Distributed Computing, v.59 n.2, p.252-279, Nov. 1999

[5] Thomas Beisel, Edgar Gabriel, Michael Resch , "An Extension to MPI for Distributed Computing on MPPs", Recent Advances in Parallel Virtual Machine and Message Passing Interface', Lecture Notes in Computer Science, 75--83, Springer (1997).

[6] Sajal K. Das, Daniel J. Harvey, Rupak Biswas, "Latency Hiding in Dynamic Partitioning and Load Balancing of Grid Computing Applications", CCGRID 2001: 347-354

[7] ENZO: CosmologicalSimulation Code. http://cosmos.ucsd.edu/enzo/.

[8] J. Faik, L. G. Gervasio, J. E. Flaherty, J. Chang, J. D. Teresco, E. G. Boman, K. D. Devine, "A model for resource-aware load balancing on heterogeneous clusters", Tech. Rep. CS-03-03, Williams College Department of Computer Science, http://www.cs.williams.edu/drum.

[9] I. Foster and C. Kesselman. The Grid: Blueprint for a New Computing Infrastructure. Morgan Kaufmann Publishers, San Francisco, California, 1999.

[10] S. Genaud, A. Giersch, F. Vivien, "Load-Balancing Scatter Operations for Grid Computing", International Parallel and Distributed Processing Symposium (IPDPS'03) April 22 - 26, 2003 Nice, France

[11] GROMACS, http://www.gromacs.org

[12] Y.F. Hu, R.J. Blake, and D.R. Emerson, "An optimal migration algorithm for dynamic load balancing", Concurrency: Practice and Experience, 10(1998):467-- 483.

[13] W. Johnston, D. Gannon, and B. Nitzberg. Grids as Production Computing Environments:

The Engineering Aspects of NASA's Information Power Grid. IEEE Computer Society Press, 1999.
[14] A. Olugbile, H. Xia, X. Liu, A. Chien from CSAG, "New Grid Scheduling and Rescheduling Methods in the GrADS Project", Workshop for Next Generation Software, Santa Fe, New Mexico, April 2004, held in conjunction with the IPDPS 2004.
[15] Z.Lan, V.Taylor, and G.Bryan, "Dynamic load balancing of samr applications on distributed systems", Proc. Of SC2001, Denver, CO, 2001.
[16] Z.Lan, V.Taylor, and G.Bryan, "A novel dynamic load balancing scheme for parallel systems", Journal of Parallel and Distributed Computing, page 1763 - 1781, 2002.
[17] Z.Lan, V.Taylor, and G.Bryan, "Exploring cosmology applications on distributed environments", Journal of Future Generation Computer Systems, pages 839--847, 2003.
[18] A. Plaat, H.E. Bal, R.F.H. Hofman, and T. Kielmann, " Sensitivity of Parallel Applications to Large Differences in Bandwidth and Latency in Two-Layer Interconnects. Future Generation Computer Systems, 17:769--782, 2001.
[19] K. Schloegel, G. Karypis, and V. Kumar, "Multilevel Diffusion Schemes for Repartitioning of Adaptive Meshes", Journal of Parallel and Distributed Computing, 47(2): 109-124,1997
[20] The NSF TeraGrid. http://www.teragrid.org.
[21] S.Vazhkudai, J. M. Schopf, I. Foster, "Predicting the Performance of Wide Area Data Transfers", Proceedings of the 16th International Parallel and Distributed Processing Symposium (IPDPS 2002), April 2002.
[22] C. Walshaw, M. Cross, and M. Everett. "Parallel Dynamic Graph Partitioning for Adaptive Unstructured Meshes", J. Par. Dist. Comp., 47(2) 102-108, 1997
[23] R. Wolskiy, "Dynamically Forecasting Network Performance Using the Network Weather Service", UCSD Technical Report TR-CS96-494, 1998
[24] Y. Yang, H. Casanova, "Multi-Round Algorithm for Scheduling Divisible Workload Applications: Analysis and Experimental Evaluation", University of California, San Diego, Dept. of Computer Science and Engineering, Technical Report CS2002-0721,2002