An Efficient Collective Communication Method for Grid Scale Networks

Kyung-Lang Park[1], Hwang-Jik Lee[1], Youn-Joo Lee[1], Oh-Young Kwon[2], Sung-Yong Park[3], Hyung-Woo Park[4], and Shin-Dug Kim[1]

[1] Dept. of Computer Science, Yonsei University
134 Shinchon-Dong, Seodaemun-Gu, Seoul 120-749, Korea
{lanx, bear22, garfield, sdkim}@parallel.yonsei.ac.kr

[2] Dept. of Computer Engineering, Korea University of Technology and Education P.O. BOX 55, Chonan, 330-600, Korea
oykwon@kut.ac.kr

[3] Dept. of Computer Science, Sogang University
1 Shinsoo-Dong, Mapo-Ku, Seoul 121-742, Korea
parksy@ccs.sogang.ac.kr

[4] Korea Institute of Science and Technology Information,
P.O. BOX 122, Yusong, Taejun, 305-806, Korea
hwpark@hpcnet.ne.kr

Abstract. This research is to design a collection of effective collective operations for the Grid scale network environment. In the Grid, several network features should be specified and adapted into the algorithmic design of collective operations. For this, we characterize the Grid and design hierarchical latency optimal tree algorithm for the MPI (message passing interface) library. The experimental results show that performance gain in performing collective operations can be achieved by around 160% compared with MPICH-G2 which is based on the flat algorithm.

1 Introduction

As the computational Grid [7, 8] is emerging, a lot of existing computational approaches are gradually changed into a new paradigm to utilize ultimately unlimited resource as a single entity. However, it's difficult to design reliable applications to be performed effectively in Grid environments. The difficulty is mainly caused by architectural features of the Grid that is configured with a large number of heterogeneous and wide area networks. In such environments, many researchers attempt to improve collective communication methods because application performance is not affected significantly by the core services of middleware but by programming and communication libraries, where collective communication methods are the central part of these libraries for efficient parallel programming environment.

Previously, a number of collective operations have been designed and implemented to be adapted and optimized for each computing environment, but not for the Grid

P.M.A. Sloot et al. (Eds.): ICCS 2003, LNCS 2660, pp. 819–828, 2003.

scale network. This research is to design effective collective operations for the Grid scale network environment. Specifically, several features from the Grid scale networks are specified and adapted into the algorithmic design of collective operations. To accomplish this goal, we designed a hierarchical latency optimal tree (HLOT) algorithm and implemented the MPI library, called MPICH-GX, by using multi-level network topology and network weather service (NWS) [12]. The proposed MPI library, MPICH-GX, shows the performance gain by around 160% when performing collective operations under our Grid testbed which comprise 8 clusters.

In Section 2, we introduce general concept of collective operations. In Section 3, we describe our advanced collective communication algorithm and implementation in detail. Section 4 shows several results of our implementation. Finally, we present the conclusions of the research in Section 5.

2 Background and Related Work

Previous MPI related studies [1,2,3,6,10,11] show that significant performance can be achieved by improving communication capability in Grid environments, especially by considering collective operations. In designing the collective operations, the key factors are to maximize the parallelism and to minimize communication cost. Based on these factors, several algorithms had been designed but these are classified into three major groups. The first group is based on binomial tree algorithm and it is focused on the parallelism of communication stages. Second one is based on minimum spanning tree (MST). They attempt to minimize communication cost by selecting the fastest link. Third one is based on the flat tree. These are methods to have reflected characteristics of wide area links and to maximize the overlapping of communication when used postal (non-blocking) model. In general, binomial tree had been chosen as an optimal algorithm when collective communication primitives were designed under MPI programming environment [9]. However, the work in [1] showed that the flat tree could provide better performance at the WAN level, because, in the flat tree, communications do not go through the WAN link more than once. Based on this wide area communication concept, flat tree algorithm was regarded as an optimal algorithm in collective communication primitives in WAN. [1,3,10]

Along with the progress of these communication algorithms, there was also significant progress in the implementation of MPI collective operations. MagPIe [2] communication library and MPICH-G2 [12] Grid-enabled library are the representatives of this research field. They implemented actual MPI library by adding the multi level communication on the previously studied algorithms that can reflect the network features. In the MagPIe system, they divide processes into two levels, i.e., WAN and LAN levels, and each process at WAN level becomes a coordinator or a sub-root manager. When any collective operation like *MPI_Bcast* starts, the root node sends the messages to the coordinators at the WAN level and each coordinator sends them to its slave nodes within its own cluster located at the LAN level. As mentioned before, the flat tree algorithm is used at the WAN level and the binomial tree algorithm is used at LAN level [3]. In the MPICH-G2, communication is divided into four levels, i.e.,

WAN, LAN, intraTCP, and vendorMPI. Also flat-tree algorithm is applied to the WAN level and the binomial tree to other levels. In such WAN environments, a sort of layered topology is chosen and communications are performed based on the order of levels [6].

However, there are still important problems to optimize collective operations in the Grids, because the Grid environments are basically constructed as heterogeneous networks with different performance ranges so that it is rather different from general wide area networks. So, it is inadequate if we define the characteristics of Grid Scale WAN as a uniform entity and apply a simple flat-tree algorithm. Thus a more adaptive algorithm is designed and implemented in this work.

3 Designing an Efficient Algorithm in Grid

3.1 Characterizing the Grid Scale Network

In designing the collective communication primitives, one of the most important features is to reflect the status of network environments under which message passing programs are performed. In early stage, binomial tree was regarded as one of the most suitable algorithms because MPI execution environments are implicitly considered as parallel machine and/or supercomputer environments that are configured with extremely high speed internal communication links. However, execution environment for the Grid scale network is extended to the workstation pools and also widely distributed clusters that are linked via external networks including the Internet. Thus, several major features for the Grid network should be clarified to be reflected to the design of collective communications.

The characteristics of grid network originates from the fact that all resources are geometrically distributed, and the following features of Grid network should be considered for designing the collective communication algorithm.

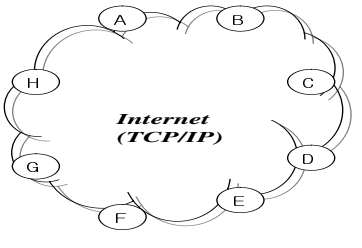
1. *Network status can be changed radically and dynamically*
2. *Latency is the most considerable parameter and not fixed as constant*

The first characteristic is due to the nature of distributed resources. The Grid comprise of the various heterogeneous networks including the Internet allowed to the mass of the people. Therefore status of the networks can be changed dynamically according to various factors such as the network traffic. We can easily experience a terrible network delay in the daytime when a lot of people are connected in networks. It shows implicitly the need of the intelligent communication algorithm which considers dynamic network status.

Second one also can be derived from the dynamic states of the network. When transferring a message from one point to another, the total communication time includes the overhead, gap, and latency [4]. The overhead can be defined as the length of time that a processor in engaged in the transmission or reception of each message. The gap is defined as the minimum time interval between consecutive message transmissions at a processor. Finally, the latency is incurred in communicating a message

containing a word from its source module to its target module, which includes the RTT (Round Trip Time), propagation delay, routing and so on. Among them, the overhead is extremely smaller than other parameters in wide area networks, so that it can be eliminated. The gap is relatively considerable. If the message size is increased, the gap is increased too because of the limitation of bandwidth. However, considering the parallel programming which comprises of only the fixed data array transmission, and not include huge data transmissions like FTP, the gap can be fixed as a relatively small constant. Thus, the most dynamic parameter is the latency and total communication delay is dominated by the latency. Surely, if we can obtain a more generalized parameter like ideal communication cost which covers all factors completely, it will be more useful. But, it also can be a naïve method which cannot be implemented.

An example shown in Figure 1 and 2 shows how the HLOT and other algorithms are performed to utilize these characteristics. Consider a case that a set of clusters is located at geometrically distributed sites as shown in Figure 1. The right side table shows the communication cost. In such an environment, if we attempt to broadcast messages by using such well-known algorithms, different results can be obtained as in Figure 2.



	A	B	C	D	E	F	G	H
A	0	400	800	420	520	500	800	150
B	430	0	200	450	700	540	700	370
C	700	250	0	500	870	850	700	600
D	400	450	500	0	500	950	1300	780
E	500	670	800	500	0	500	700	520
F	580	550	850	800	500	0	280	480
G	920	700	700	1260	720	300	0	200
H	300	520	600	800	550	320	200	0

Fig. 1. Latency matrix of virtual organizations in Grid scale network.

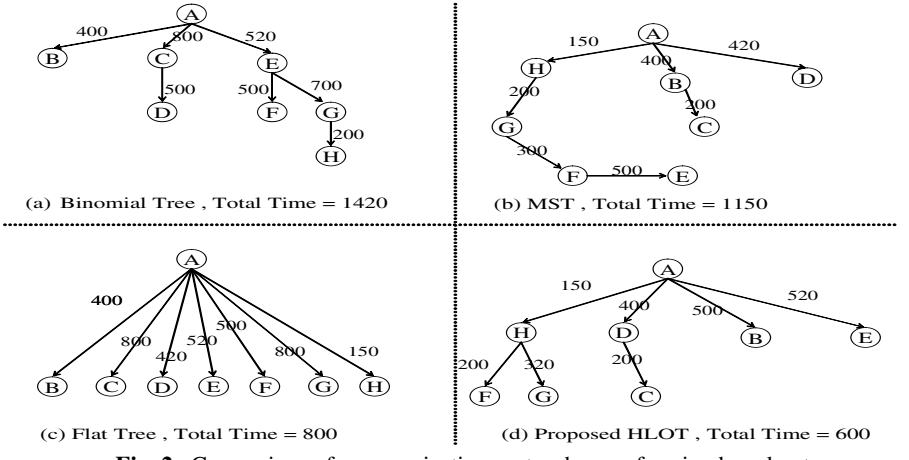


Fig. 2. Comparison of communication costs when performing broadcast

Each algorithm has its own unique characteristics and advantage, but the proposed HLOT (Hierarchical Latency Optimal Tree) algorithm shows the most effective performance in extremely wide area network such as the Grid. We will explain about this HLOT algorithm in more detail in the next section. First candidate is a binomial tree algorithm. We already mentioned that it is focused on minimizing the number of send/receive stages. But, in Grid network, overhead is extremely small so that all transmissions from one node can be treated as just one stage using non-blocking send. Therefore, the merit of binomial tree cannot be effective in this case. As shown in Figure 2, the binomial tree includes a path connected by A, E, G, and H, which takes 1420 time units. The second one is a minimum spanning tree (MST) algorithm which selects only the lowest latency link one by one. It can be latency optimal in Grid network but such a long, inefficient path can be occurred. In Figure 2, a long path of A, H, G, F, and E is determined for broadcasting so that overall communication time can be 1150. To improve the efficiency of MST, that path should be removed. The third one is a flat tree algorithm which can maximize the merit of non-blocking send. It can broadcast in only one stage by sending all messages from the root to others. But, if there are one or more high latency links from root, its performance can be degraded significantly. Thus, long latency occurred between A and G, but flat tree algorithm can't avoid that link. Finally, the proposed HLOT algorithm can complement those disadvantages of other algorithms. Basically, the root node is in charge of transferring messages and communication links with long latencies from the root node are eliminated. In Figure 2, the HLOT shows 600 time units as the lowest communication path for broadcasting message.

3.2 Hierarchical Latency-Optimal Tree (HLOT) Algorithm

In this section, we will present the detailed design specification and analysis of the HLOT algorithm. The HLOT algorithm is based on Prim's algorithm and created by adapting additional two principles basically. The first one is the latency-optimal feature. Namely the algorithm is based on the latency information as an important feature rather than considering any other parameters in the Grid environment. Thus communication links consisting of an optimal latency are selected. The next important feature is to consider direct links first. For the collective operations, using direct links from the root node can be effective due to the characteristics of wide area network. The basic algorithm is shown in Figure 3.

The HLOT algorithm is designed by taking advantage of structural merits of the flat tree and the minimal spanning tree. Especially, the HLOT can provide the effect of non-blocking send operations as in the flat tree, but it also can avoid some high latency links by selecting latency optimal links. Performance evaluation of the HLOT algorithm is appeared in Section 5, but we first analyze the algorithm via LogP model [4] that is commonly used as a performance model for message passing system. LogP is a modeling technique using four parameters such as latency, overhead, gap, and processors. The time to broadcast can be described as following LogP model.

Variables :
Y : a set of selected nodes, V : an universal set of nodes
W[][] : 2-dimensional latency array, e(c,n): the edge from c to n
D[n] : the sum of latencies for the selected path from the root to node n
Output :
F : a set of selected edges,
Algorithm steps :

```
F = Ø ;
Y = { root };
while (V-Y • Ø ) {     // repeat until finding the path to all nodes
    choose fastest link e(c,n);  // (c is an element of Y and n is one of V-Y)
    if( D[n] is higher than W[0][n] )
            continue;          // try again
    add c to Y;
    add e(c,n) to F;
    Update D[n] }
```

Fig. 3. HLOT algorithm.

First, T_B that is the total time to broadcast a particular message can be expressed by selecting maximum value of completion times required for all processes participating in broadcasting.

$$T_B = \sum_{i=0}^{size-1} Max\{T_{pi}\}, \text{ where size = | V |} \tag{1}$$

T_{Pi} means the completion time of the *i*-th process expressed as the following equation.

$$T_{Pi} = \begin{cases} T_{Pj} + (c-1)(T_{SO} + T_G) + L_{ji} + T_{RO} (i \neq 0) \\ 0 (i = 0) \end{cases} \tag{2}$$

where *j* is the parent's index of the *i*-th process, T_{Pj} is the delivery time from the root to j, c is the number of child nodes of *j*, and T_{SO}, T_G, T_{RO} are the time of sending overhead, gap, and receiving overhead in order. That is, the time to receive a message from the *i*-th process can be obtained as the summation of the time delivered to the parent and the time to be transferred from the parent. In the case of flat tree algorithm, T_{pj} becomes zero all the time because the root node is the parents to all the processes. Also L_{ji} becomes L_{0i} as the latency directly from the root from the *i*-th process. As mentioned, the overhead involved in message transfer is extremely smaller than the completion time in Grid environments. Assume fixed size data transfer, gap and overhead parameters can be eliminated. Thus the broadcasting time T_B can be represented as:

$$\text{Flat Tree: } T_B = Max\{L_{0i}\} \tag{3}$$

$$\text{Others: } T_B = Max\{L_j + L_{ji}\} \tag{4}$$

where *Lj* is the sum of latency in paths form 0 to *j*. The biggest value of latency from the root to all processes (L_{0i}) influence the whole performance in flat tree algorithm. But, in others such as MST and HLOT, the biggest value in sum of the latency paths coming to the parent (L_j) and the latency from the parent to child (L_{ji}) influence whole performance. In designing the HLOT algorithm, we assume that a link is selected when the sum of L_j and L_{ji} is smaller than that of L_{0i}, so that HLOT algorithm is faster than flat tree always. But, it is not always true in other algorithms such as MST, binomial, and so on. On the contrary, by the use of several long distance links, performance becomes worse than that of flat tree.

3.3 Implementation

In general, some restrictions exist to design and implement the proposed algorithm as the collective communication primitives in the MPI library. HLOT algorithm also meets two important restrictions for implementation. The first one is how to collect latency information. The overhead to measure the latency of all nodes occurs necessarily. Therefore, we need a mechanism to measure the correct latency and not to countervail the efficiency of the HLOT algorithm. The second one is the overhead of HLOT algorithm itself. The process of finding an optimal latency path is based on the greedy approach. The HLOT algorithm show more than $O(n^2 \log n)$ complexity. Thus, a policy to determine the status of nodes required for the HLOT algorithm is required. Two techniques, the NWS (Network Weather Service) [12] and the multi-level communication mechanism [6] are chosen for this goal.

The NWS is a component that can supply network state between linked nodes. The NWS also provides APIs (Application Programming Interface) so that users can program with NWS directly. To collect latency information, these APIs are modified into the MPI Library. Until now, we utilize latency information of one specific point, but can utilize to compose topology by intelligence using continuous statistical data and prediction.

Multi-level communication is a method that can improve the performance by separating the given network into several layers according to its characteristics. It can select the suitable algorithms for each layer. We also leverage the concept of multi-level communication by adapting our algorithms into MPI library efficiently.

MPICH-G2 that is widely used as the Grid-enabled MPI library divides the network into four layers. But the Grid implicitly means the use of extremely distributed computing and network resources, and four-level topology is a very restrictive communication architecture. Also, other three levels except the WAN level include only local area without considering the WAN level. Consequently, the WAN level should be classified into more levels to cope with the various network environments in Grids. Therefore we divide the WAN into two levels, called as light-WAN and heavy-WAN, and use the flat tree algorithm to the light-WAN level which includes relatively stable and uniform links and apply the HLOT algorithm to the heavy-WAN level which includes various heterogeneous links with four characteristics mentioned in Section 3. A user can divide these levels by using explicit parameters in the RSL file or it can be performed automatically by using latency information obtained by NWS. Figure 4

shows an example of abstract diagram constructed by the five-level topology with HLOT algorithm. In Figure 4, the root process broadcasts a message to sub-root by using HLOT. Thus a heavy latency link can be avoided in heavy-WAN, drawn as the dotted line.

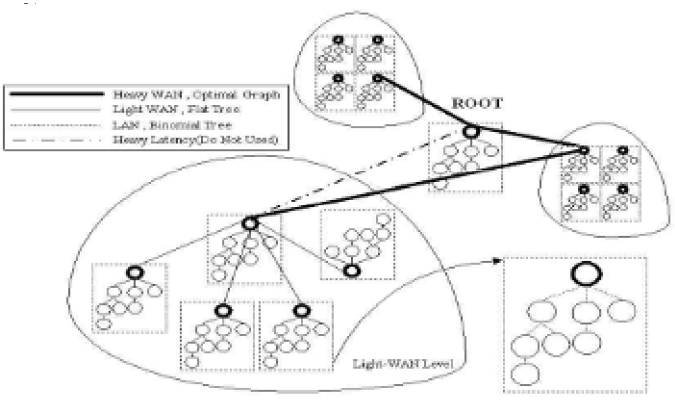


Fig. 4. Multi-level communication with HLOT algorithm.

4 Experimental Result

To evaluate the impact of our HLOT algorithm and its MPI implementation, we will show the experimental results in a practical Grid environment. Our testbed consists of 8 clusters of 1GHz/512MB Pentium machines connected by general purpose Internet and located at 4 distributed organizations – Yonsei, Ajou, KUT and KISTI. Our workload comprises a *simple MPI_Bcast* using 2 byte message size which is a representative collective operation in MPI Libraries, and it can clearly show the behavior of implemented MPI system. There are several accurate measurement techniques, but we execute a program by once for each sample to make the most independent events. It can avoid pipelining effects absolutely. To obtain 99% confidence, we observe the 130 samples which are shown in Figure 5. In this figure, the dotted line means the samples of MPICH-G2 which uses the 4 level topology and flat algorithm and the solid line means the samples of MPICH-GX (eXtension) which uses the extended 5 level topology and the HLOT algorithm.

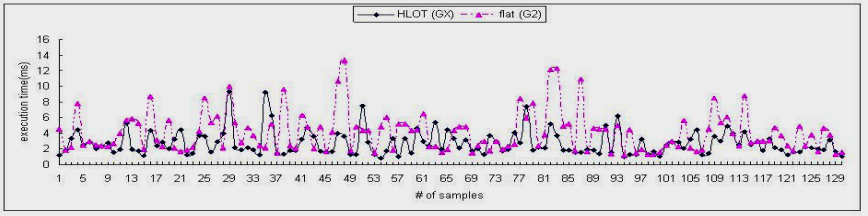


Fig. 5. Observed samples of MPI_Bcast

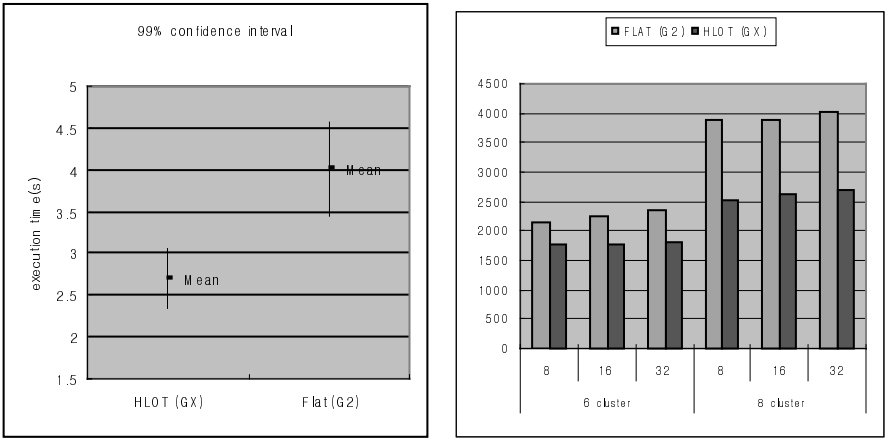


Fig. 6. (a) Approximate visual test between the HLOT (GX) and the flat tree (G2). (b) Average completion time of *MPI_Bcast* in different clusters

We repeated the same experiments by using different numbers of clusters, processes, and message sizes. As shown in Figure 6 (b), if the number of clusters is increased, performance gain increases more significantly because a lot of high latency links are involved. In the 8 cluster system with 32 processes, we can obtain the maximum rate of improvement. In Figure 7, if the size of messages becomes bigger, performance improvement becomes a little larger because the disadvantage caused by using high latency link is dependant on the message size. Message size can affect other parameters like gap. But, we are focused on the latency parameter because latency is the most effective parameter as we mentioned in Section 3. The relation between message size and other parameters is described in [2] by detail.

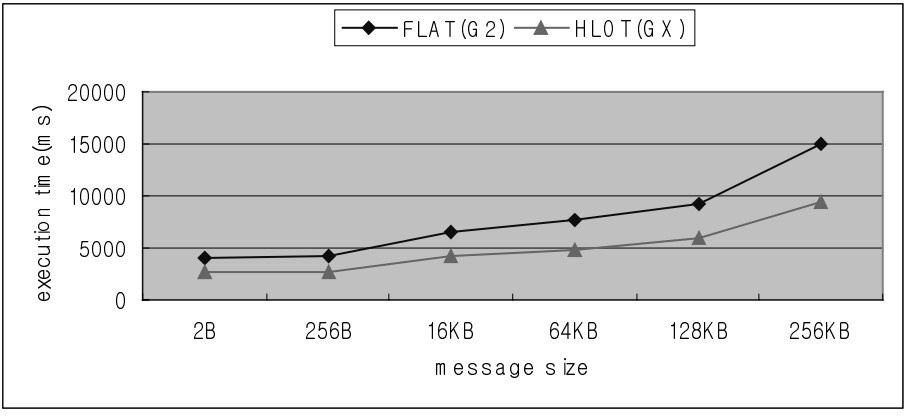


Fig. 7. MPI_Bcast with different message sizes.

5 Conclusion

The goal of this research is to improve the performance of collective communication method in the Grid scale networks. Current various programming libraries attempt to optimize the communication performance, but they still don't consider the various characteristics of the Grid network. So, we design the HLOT algorithm which is based on the features of Grid aggressively and implement the MPI library, MPICH-GX by using the proposed algorithm with two methods, i.e., multi level topology and network weather service. Thus the performance improvement can be achieved by around 160% compared with MPICH-G2 in our testbed which is distributed in 4 locations and comprise 8 clusters. Here, the basic concept of wide area collective communications based on the flat tree should be changed by considering the various network characteristics.

References

1. M. Bernaschi and G.Iannello. Collective Communication Operations: Experimental Results vs. Theory, Concurrency : Paractice and Experience, 10(5), (1998), 359–386.
2. T. Kielmann, H. E. Bal and S. Gorlatch, Bandwidth-efficient Collective Communication for Clustered Wide Area Systems, In proc. International Parallel and Distributed Processing Symposium, Mexico, (2000), 492–499.
3. T. Kielmann, R. F. H. Hofman, H. E. Bal, A. Plaat, and R. A. F. Bhoedjang, MagPIe: MPI's Collective Communication Operations for Clustered Wide Area Systems, In Proc. Symposium on Principles and Practice of Parallel Programming , Atlanta, GA, 5 (1999), 131–140.
4. D. Culler, R. Karp, D. Patterson, A. Sahay, K.E. Schauers, E. Santos, R.Subramonian and T. von Eicken. LogP: Towards a Realistic Model of Parallel Computation, In Proc. Symposium on Principles and Practice of Parallel Programming, CA , 5 (1993) , 1–12.
5. I. Foster and N. Karonis, A grid-enabled MPI: Message passing in heterogeneous distributed computing systems, In Proc. Supercomputing '98, 11 (1998).
6. N. Karonis, B. de Supinski, I. Foster, W. Gropp E. Lusk, and J. Bresnahan, Exploiting hierarchy in parallel computer networks to optimize collective operation performance, In Proc. International Parallel and Distributed Processing Symposium, (2000).
7. I. Foster and C. Kesselman, eds. The GRID : Blueprint for a New Computing Infrastructure, Morgan Kaufmann, (1998).
8. I. Foster and C. Kesselman, "Globus: A metacomputing infrastructure toolkit", International Journal of Supercomputer Applications, 11(2), (1997), 115–128.
9. Message Passing Interface Forum, MPI: A Message-Passing Interface standard, International Journal of Supercomputer Applications, 8(3/4), (1994), 165–414.
10. MPICH-G2, Online at http://www3.niu.edu/mpi/.
11. P.B. Bhat, C.S. Raghavendra, and Viktor K. Prasanna, Efficient Collective Communication in Distributed Heterogeneous Systems, International Conference on Distributed Computing Systems, 5 (1999).
12. Network Weather Service, Online at http://nws.cs.ucsb.edu/.