TOTAL EXCHANGE PERFORMANCE PREDICTION ON GRID ENVIRONMENTS

modeling and algorithmic issues

Luiz Angelo Steffenel and Emmanuel Jeannot
Université Nancy 2 / LORIA - AlGorille Team*
LORIA - Campus Scientifique - BP 239
54506 Vandoeuvre-lès-Nancy Cedex
France
Luiz-Angelo.Steffenel@univ-nancy2.fr
Emmanuel.Jeannot@loria.fr

Abstract One of the most important collective communication patterns used in scientific applications is the *complete exchange*, also called *All-to-All*. Although efficient algorithms have been studied for specific networks, general solutions like those available in well-known MPI distributions (e.g. the MPI_Alltoall operation) are strongly influenced by the congestion of network resources. In this paper we address the problem of modeling the performance of *Total Exchange* communication operations in grid environments. Because traditional performance models are unable to predict the real completion time of an All-to-All operation, we try to cope with this problem by identifying the factors that can interfere in both local and distant transmissions. We observe that the traditional MPI_Alltoall implementation is not suited for grid environments, as it is both inefficient and hard to model. We focus therefore in an alternative algorithm for the total exchange redistribution problem. In our approach we perform communications in two different phases, aiming to minimize the number of communication steps through the wide-area network. This reduction has a direct impact on the performance modeling of the MPI_Alltoall operation, as we minimize the factors that interfere with wide-area communications. Hence, we are able to define an accurate performance modeling of a total exchange between two clusters.

Keywords: MPI, all-to-all, total exchange, network contention, performance modeling, computational grids, personalized many-to-many communications

*UMR 7503 - CNRS, INPL, INRIA, UHP, Nancy 2

1. Introduction

One of the most important collective communication patterns for scientific applications is the *total exchange* [1], in which each process holds n different data items that should be distributed among the n processes, including itself. An important example of this communication pattern is the All-to-All operation, where all messages have the same size *m*.

Generally, most All-to-All algorithms from well-known MPI distributions rely on direct point-to-point communications among the processes. Because all these communications are started simultaneously, the communication performance is strongly influenced by the saturation of network resources and subsequent loss of packets - the network contention. Further, when working in a grid, we must also face problem related to the heterogeneous communication environment, which behaves differently if message exchange are made locally or remotely.

In this paper we study different approaches to model the performance of the All-to-All collective operation in grid environments. Performance prediction can be extremely helpful on the development of application performance prediction frameworks such as PEMPIs [2], but also in the optimization of grid-aware collective communications (e.g.: LaPIe [3] and MagPIe [4]). We demonstrate that traditional algorithms for the MPI_Alltoall operation are hard to model because of the combined complexity of both local-area contention and wide-area latency.

This paper is organized as follows: Section 2 presents the problem of the total exchange and the challenges we face in a grid environment. Section 3 discusses the existing approaches to introduce the network contention in the performance models for the MPI_Alltoall operation. Section 4 extends the performance prediction problem to a grid environment. We propose a new algorithmic approach that helps minimizing the contention impact, and we validate its performance modeling against experimental data obtained on a grid network. Finally, Section 5 presents some conclusions and the future directions of our work.

2. Problem of Total Exchange between Two Clusters

We consider the following architecture (see Figure 1). Let there be two clusters $\mathcal{C}_1$ and $\mathcal{C}_2$ with respectively n_1 nodes and n_2 nodes. A network, called a backbone, interconnects the two clusters. We assume that a cluster use the same network card to communicate to one of its node or to a node of another cluster. Based on that topology inter cluster communications are never faster than communication within a cluster.

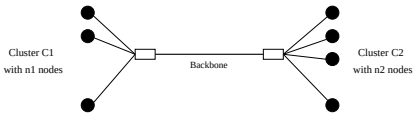

Figure 1. Architecture for the redistribution problem

Let us suppose that an application is running and using both clusters (for example, a code coupling application). One part of the computation is performed on cluster $\mathcal{C}_1$ and the other part on cluster $\mathcal{C}_2$. During the application, data must be exchanged from $\mathcal{C}_1$ to $\mathcal{C}_2$ using the *all-toall* pattern. *Alltoall* (also called total exchange) is defined in the MPI standard. It means that every node has to send some of its data to all the other nodes. Here we assume that the data to be transfer is different for each receiving node (if the data is the same, the routine is called an *allgather* and is less general that the studied case). Moreover we assume that the size of the data to exchange is the same for every pair of nodes (the case where the size is different is implemented by the *alltoallv* routine: it is more general than our case and will be studied in a future work). Altogether, this means that we will have to transfer $(n_1 + n_2)^2$ messages over different network environments. The data of all these messages are different but the size of the messages are the same and is given and called m (in bytes). Several MPI libraries (OpenMPI, MPICH2, etc.) implement the *alltoall* routine assuming that all the nodes are on the same clusters, which means that all communications have the same weight. However, in our case, some messages are transferred within a cluster (from a node of $\mathcal{C}_1$ to a node of $\mathcal{C}_1$ or from $\mathcal{C}_2$ to $\mathcal{C}_2$) or between the two clusters. In the first case, bandwidth and latency are faster than in the second case. Therefore, we need different tools to model the overall performance.

3. Modeling Network Contention

In the *All-to-All* operation, every process holds $m \times n$ data items that should be equally distributed among the n processes, including itself. The intensive communication among the processes can easily saturate the network, degrading the communication performance. Indeed, Chun [5] demonstrated that the overall execution time of intensive exchange collective communications is strongly dominated by the network contention and congestive packet loss, two aspects that are not easy to quantify. As a result, a major challenge on modeling the All-to-All operation in local-area networks is to represent the impact of network contention.

Unfortunately, most communication models like those presented by Christara *et al.* [1] and Pjesivac-Grbovic *et al.* [6] do not take into account the potential impacts of network contention. These works usually represent the All-to-All operation as parallel executions of the *personalized one-to-many* pattern [7], as presented by the linear model below, where α is the start-up time (the latency between the processes), $\frac{1}{\beta}$ is the bandwidth of the link, m represents the message

size in bytes and n corresponds to the number of processes involved in the operation:

$$T = (n-1) \times (\alpha + \beta m) \tag{1}$$

To correct the performance predictions, Bruck [8] suggested the use of a *slowdown factor*. Similarly, Clement *et al.* [9] introduced a technique that suggested a way to account contention in shared networks such as non-switched Ethernet, consisting in a contention factor γ proportional to the number of process. The use of a contention factor was supported by the work of Labarta et *al.* [10], that intent to approximate the behavior of the network contention by considering that if there are *m* messages ready to be transmitted, and only *b* available buses, then the messages are serialized in $\lceil \frac{m}{b} \rceil$ communication waves.

A slightly different approach was followed by Chun [5], who consider the contention as a component of the communication latency, resulting in the use of different latency values according to the message size. One drawback, however, it that this model does not take into account the number of messages passing in the network nor the link capacity, which is related to the occurrence of network contention.

3.1 Performance modeling in homogeneous clusters

To cope with this problem and to model the impact of contention on the All-to-All operation in cluster environments, we presented in [11] an approach inspired in the work from Clement *et al.* [9]. In our approach, the network contention depends mostly on the physical characteristics of the network (network cards, links, switches). Consequently, we can define a contention ratio γ that bounds the theoretical model from Equation 1 and the real performance of the network.

Our method differs from previous one by considering that communication times are not linear regarding the message size. Indeed, we observed that the communication time presents a non-linear behavior according to some factors such as MPU message segmentation, MPI transmission policy and switches maximum interconnection bandwidth.

Therefore, we augment the *contention ratio* model with a new parameter δ, which depends on the number of processes but also on a given message size M, as seen below. As a consequence, we are able to associate different equations (linear and affine) in order to help defining a more realistic performance model for the MPI_Alltoall operation in a given network, as illustrated in Figure 2.

$$T = \begin{cases} (n-1) \times (\alpha + m\beta) \times \gamma & if\ m < M \\ (n-1) \times ((\alpha + m\beta) \times \gamma + \delta) & if\ m \geq M \end{cases} \tag{2}$$

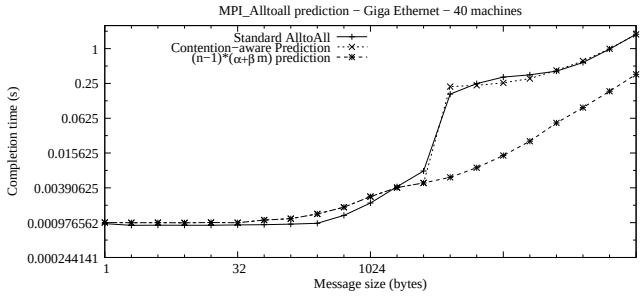

Figure 2. Measured and predicted performance for the standard MPI_Alltoall in a Gigabit Ethernet network

4. Performance Modeling on Grid Environments

As the previous model allows a quite accurate representation on the performance of local-area networks (see [11]), our first approach would be to estimate the communication time by composing both local (contention-aware) and remote communications.

Unfortunately, this simple strategy fails to represent the operation of the MPI_Alltoall in a grid. Hence, Figure 3 presents the completion time of the MPI_Alltoall implementation from OpenMPI in a grid with two clusters of 30 machines each. As stated above, we try to predict the communication performance by individually representing local and remote communication costs. To predict the performance of the local network (subjected to contention), we use $\gamma = 2.6887$ and $\delta = 0.005039$ as the contention signature of each local network (both clusters have similar characteristics under contention).

Actually, we observe that the local-area part plays a small role in the overall execution time, compared to the wide-area communication cost. Of course, one could try to define additional parameters for the wide-area communications, but the final model would be too complex to be useful in real situation. Instead, we addressed this problem by redefining the All-to-All problem against the challenges that characterize a grid environment.

4.1 Minimizing the impact of contention on the backbone

When dealing with wide-area networks, the most important factor to be considered is the time a message takes to be delivered. Indeed, in addition to the geographical distance, message are subjected to network protocols heterogeneity, message routing and transient interferences on the backbone.

Actually, popular algorithms for collective communications on grids (such as the ones implemented in PACX MPI [12] and MagPIe [4]) try to minimize communications over the wide-area network by defining a single coordinator in every cluster, which participates in the inter-cluster data transfers across the

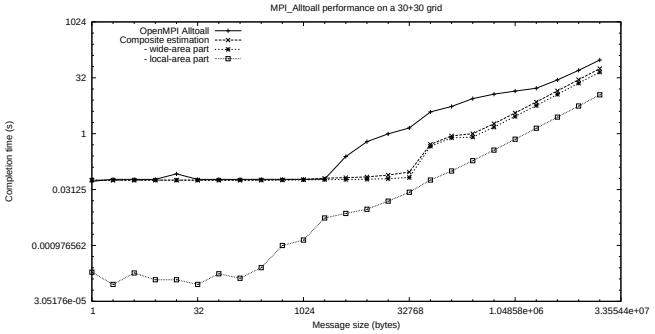

Figure 3. Measured and predicted performance for the standard MPI_Alltoall in a grid

wide-area backbone. By minimizing the number of WAN communication steps, we reduce the probability of inducing contention and accumulating transmission delays on the messages.

However, a single communication between each cluster is an approach inappropriate for the MPI_Alltoall operation. First, it induces additional communication steps to/from the cluster coordinator, which becomes a bottleneck. Second, this approach is not optimal concerning the usage of the wide-area bandwidth, as wide-area backbones are designed to support simultaneous transfers and simultaneous transfers [13]. Hence, in order to improve the performance in a WAN, we need to change the MPI_Alltoall algorithm strategy.

4.2 The $\mathcal{LG}$ algorithm

To cope with this problem, we try to minimize wide-area communication steps in a different way. Actually, most of the complexity of the All-to-All problem resides on the need to exchange *different* messages through different networks (local and distant). The traditional implementation of the MPI_Alltoall operation cannot differentiate these networks, leading to poor performances. However, if we assume that communications between clusters are slower than intra-clusters ones, it might be useful to collect data in the local level before sending it in parallel through the backbone, in a single communication step.

As a consequence, we propose in [14] a grid-aware solution which performs on two phases. In the first phase only local communications are performed. During this phase the total exchange is performed on local nodes on both cluster and extra buffers are prepared for the second (inter-cluster) phase. During the second phase data are exchanged between the clusters. Buffers that have been prepared during the first phase are sent directly to the corresponding nodes in order to complete the total exchange.

More precisely, our algorithm works as follow. Without loss of generality, let us assume that cluster $\mathcal{C}_1$ has less nodes than $\mathcal{C}_2$ ($n_1 \leq n_2$). Nodes are numbered from 0 to $n_1 + n_2 - 1$, with nodes from 0 to $n_1 - 1$ being on $\mathcal{C}_1$ and

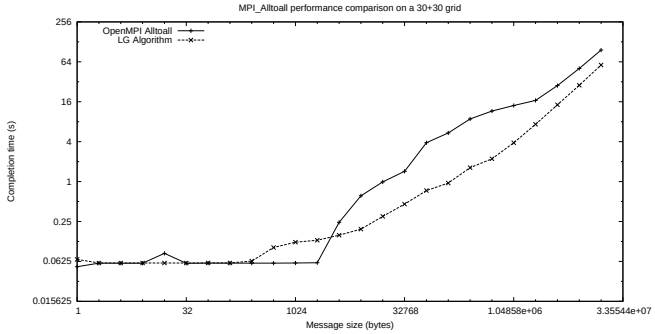

Figure 4. Performance comparison between OpenMPI and $\mathcal{L}G$ algorithms

nodes from n_1 to $n_1 + n_2 - 1$ being on cluster $\mathcal{C}_2$. We call $\mathcal{M}_{i,j}$ the message (data) that has to be sent from node i to node j. For instance, the algorithm proceeds in two phases:

First phase During the first phase, we perform the local exchange: Process i sends $\mathcal{M}_{i,j}$ to process j, if i and j are on the same cluster. Then it prepares the buffers for the remote communications. On $\mathcal{C}_1$ data that have to be send to node j on $\mathcal{C}_2$ is first stored to node $j \ mod \ n_1$. Data to be sent from node i on $\mathcal{C}_2$ to node j on $\mathcal{C}_1$ is stored on node $\lfloor i/n_1 \rfloor \times n_1 + j$.

Second phase During the second phase only n_2 inter-cluster communications occurs. This phase is decomposed in $\lceil n_2/n_1 \rceil$ steps with at most n_1 communications each. Steps are numbered from 1 to $\lceil n_2/n_1 \rceil$ During step s node i of $\mathcal{C}_1$ exchange data stored in its local buffer with node $j = i + n_1 \times s$ on $\mathcal{C}_2$ (if $j < n_1 + n_2$). More precisely i sends $\mathcal{M}_{k,j}$ to j where $k \in [0, n_1]$ and j sends $\mathcal{M}_{k,i}$ to i where $k \in [n_1 \times s, n_1 \times s + n_1 - 1]$.

As our algorithm minimizes the number of inter-cluster communications between the clusters, we need only $2 \times \max(n_1, n_2)$ messages in both directions (against $2 \times n_1 \times n_2$ messages in the traditional algorithm). For instance, the exchange of data between two clusters with the same number of process will proceed in one single communication step of the second phase. Our algorithm is also wide-area optimal since it ensures that a data segment is transferred only once between two clusters separate by a wide-area link. Additionally, wide-area transmissions pack several messages together, reducing the impact of transient interferences on the backbone. Hence, Figure 4 presents a comparison between the traditional algorithm used by OpenMPI and the $\mathcal{L}G$ algorithm. We observe that $\mathcal{L}G$ improves the performance of the MPI_Alltoall operation, reaching over than 50% of performance improvement comparing to the traditional strategy.

4.3 Modeling approach

As shown above, the algorithm we propose to optimize All-to-All communications in a grid environment rely on the relative performances of both local

and remote networks. Indeed, we extend the total exchange among nodes in the same cluster in order to reduce transmissions through the backbone.

This approach has two consequences for performance prediction: First, it prevents contention in the wide-area links, which are hard to model. Second, the transmission of messages packed together is less subjected to network interferences. For instance, we can design a performance model by composing local-area predictions obtained with our contention ratio model and wide-area predictions that can be easily obtained from traditional methods. Hence, an approximate model would consider the following parts, where $\mathcal{T}_{C_n}$ corresponds to Equation 2:

$$T = max(T_{\mathcal{C}_1}, T_{\mathcal{C}_2}) + \lceil n_2/n_1 \rceil \times (\alpha_w + \beta_w \times m \times n_1) \quad (3)$$

4.4 Experimental validation

To validate the algorithm we propose in this paper, this section presents our experiments to evaluate the performance of the MPI_Alltoall operation with two clusters connected through a backbone.

These experiments were conducted over two clusters of the Grid'5000 platform [1], one located in Nancy and one located in Rennes, approximately 1000 Km from each other. Both clusters are composed of identical nodes (dual Opteron 246, 2 GHz) locally connected by a Gigabit Ethernet network and interconnected by a private backbone of 10 Gbps. All nodes run Linux, with kernel 2.6.13 and OpenMPI 1.1.4. The measures were obtained with the *broadcast-barrier* approach [15].

To model the communication performance of both *inter-cluster* and *intra-cluster* communications we use the *parameterised LogP* model (*pLogP*) [4]. The *pLogP* parameters for both local and distant communications were obtained with the method described in [16]. To model the contention at the local level we used $\gamma = 2.6887$ and $\delta = 0.005039$ for $M >= 1KB$, parameters obtained from the method of the least squares as described in [11].

Therefore, in Figure 5 we compare the performance predictions obtained with Equation 3 against the effective completion time of the $\mathcal{L}G$ algorithm. We observe that prediction fit with a good accuracy to the real execution times, which is not possible with the traditional MPI_Alltoall algorithm. Indeed, the new algorithm minimizes the impact of distant communications, concentrating the contention problems at the local level. Because we are able to predict the performance of local communications even under contention, we can therefore establish an accurate performance model adapted to grid environments.

[1] *http://www.grid5000.org/*

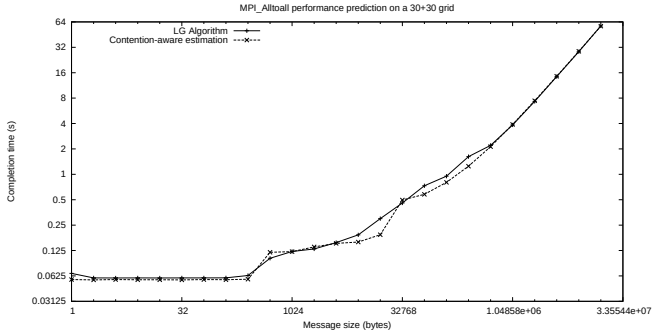

Figure 5. Performance predictions for the $\mathcal{L}G$ algorithm

5. Conclusions and Future Works

In this paper we address the problem of modeling the performance of *Total Exchange* communication operations in grid environments. Because traditional performance models are unable to predict the real completion time of an All-to-All operation, we try to cope with this problem by identifying the factors that can interfere in both local and distant transmissions. We observe that the traditional MPI_Alltoall implementation is not suited for grid environments, as it is both inefficient and hard to model. We focus therefore in an alternative algorithm for the total exchange redistribution problem. In our approach we perform communications in two different phases, aiming to minimize the number of communication steps through the wide-area network. This reduction has a direct impact on the performance modeling of the MPI_Alltoall operation, as we minimize the factors that interfere with wide-area communications.

In our future works we plan to extend the model to handle more complex distributions. First, we would like to consider achieving efficient *alltoall* communications with more than two clusters. This would allow efficient communications on general grid environments. Second, we would like to explore the problem of total exchange redistribution when messages have different sizes. This problem, represented by the *alltoallv* routine, is more general than our case and does requires adaptive scheduling techniques.

Acknowledgments

Experiments presented in this paper were carried out using the Grid'5000 experimental testbed, an initiative from the French Ministry of Research through the ACI GRID incentive action, INRIA, CNRS and RENATER and other contributing partners (see https://www.grid5000.fr).

References

[1] C. Christara, X. Ding and Ken Jackson. An efficient transposition algorithm for distributed memory computers. *Proc. of the High Performance Computing Systems and Applications*, pages 349-368, 1999.

[2] E. T. Midorikawa, H. M. Oliveira and J. M. Laine. PEMPIs: A New Metodology for Modeling and Prediction of MPI Programs Performance. *Proc. of the SBAC-PAD 2004*, IEEE Computer Society/Brazilian Computer Society, pages 254-261, 2004.

[3] L. A. Steffenel and G. Mounie. Scheduling Heuristics for Efficient Broadcast Operations on Grid Environments. *Proc. of the Performance Modeling, Evaluation and Optimization of Parallel and Distributed Systems Workshop - PMEO'06 (associated to IPDPS'06)*, IEEE Computer Society, April 2006.

[4] T. Kielmann, H. Bal, S. Gorlatch, K. Verstoep and R. Hofman. Network Performance-aware Collective Communication for Clustered Wide Area Systems. *J. Parallel Computing* **27**(11):1431-1456, 2001.

[5] A. T. T. Chun. Performance Studies of High-Speed Communication on Commodity Cluster. *PhD. Thesis*, University of Hong Kong, 2001.

[6] J. Pjesivac-Grbovic, T. Angskun, G. Bosilca, G. E. Fagg, E. Gabriel and J. J. Dongarra. Performance Analysis of MPI Collective Operations. *Proc. of the Wokshop on Performance Modeling, Evaluation and Optimisation for Parallel and Distributed Systems (PMEO), in IPDPS 2005*, 2005.

[7] S. L. Johnssonn and C-T. Ho. Optimum Broadcasting and Personalized Communication in Hypercubes. *IEEE Transactions on Computers* **38**(9):1249-1268, 1989.

[8] J. Bruck, C-T. Ho, S. Kipnis, E. Upfal and D. Weathersby. Efficient algorithms for all-to-all communications in multiport message-passing systems. *IEEE Transactions on Parallel and Distributed Systems* **8**(11):1143-1156, 1997.

[9] M. Clement, M. Steed and P. Crandall. Network performance modelling for PM clusters. *Proc. of Supercomputing*, 1996.

[10] J. Labarta, S. Girona, V. Pillet, T. Cortes and L. Gregoris. DiP: A parallel program development environment. *Proc. of the 2nd Euro-Par Conference*, vol. 2, pages 665-674, 1996.

[11] L.A. Steffenel. Modeling Network Contention Effects on AlltoAll Operations. in *Proc. of the IEEE Conference on Cluster Computing (CLUSTER 2006)*, September 2006.

[12] E. Gabriel, M. Resch, T. Beisel, and R. Keller. Distributed computing in a heterogeneous computing environment. In *Proc. of the Euro PVM/MPI 1998*. LNCS 1497, pages 180-187, 1998.

[13] H. Casanova. Network modeling issues for grid application scheduling. *International Journal of Foundations of Computer Science* **16**(2):45-162, 2005.

[14] E. Jeannot and L. A. Steffenel. Fast and Efficient Total Exchange on Two Clusters. Submitted to EuroPar'07 - 13th International Euro-Par Conference European Conference on Parallel and Distributed Computing.

[15] B. Supinski, N. Karonis. Accurately Measuring MPI Broadcasts in a Computational Grid. In *8th IEEE International Symposium on High Performance Distributed Computing (HPDC'99)*, 1999.

[16] T. Kielmann, H. Bal, and K. Verstoep. Fast measurement of LogP parameters for message passing platforms. In *4th Workshop on Runtime Systems for Parallel Programming*. LNCS Vol. 1800, pages 1176-1183, 2000.