A Performance Model of Many-to-One Collective Communications for Parallel Computing

Alexey Lastovetsky, Maureen O'Flynn
School of Computer Science and Informatics
Belfield, Dublin 4, Ireland
alexey.lastovetsky@ucd.ie, maureen.oflynn@ucd.ie

Abstract

This paper presents a performance model of Many-to-One collective communications for MPI platforms on a switched Ethernet network. The model is based on empirical findings from observation of Many-to-One operations over a wide range of message sizes. The model reflects a significant increase in the execution time for medium-sized messages, persistently observed for different parallel platforms and MPI implementations and not reflected in traditional communication performance models. We also demonstrate that the use of the model can significantly improve the performance of parallel applications, intensively using Many-to-One communications.

Key Words
High performance computing, Parallel computing, Cluster computing, Communication networks, Performance modeling, MPI, Collective communications, Many-to-One communications

1. Introduction

In our research, we aim at an accurate performance model of communications of parallel MPI applications on a heterogeneous cluster based on a switched Ethernet network. The ultimate purpose of the model is to predict the execution time of any given combination of communication operations.

In our previous work [1], we have proposed a performance model covering a single point-to-point communication, parallel combinations of independent point-to-point communications, and collective communications of the One-to-Many type. In this paper, we extend the model to Many-to-One communications.

The performance model of Many-to-One communication operations is based on observations made during extensive experiments on different platforms and implementations of MPI. In the experiments, we used MPI communications in standard mode. The model is a simple linear representation for small messages, but as the message size is increased there is a distinct region of nonlinear response. We observed from experiments sharp escalations of the execution time for medium-sized messages to one of a small number of discrete levels. Our model provides the number of escalation levels, the value of each level and an estimation of the probability of escalation of the execution time to each of the values. For large messages, the escalations cease, and a linear relationship between the execution time and the message size resumes again.

The rest of the paper is structured as follows. Section 2 outlines related work. In Section 3, we discuss observations made during extensive experiments on different platforms and MPI implementations. Section 4 proposes a performance model of Many-to-One communications summarizing the observations. Section 5 describes how to determine the value of parameters of the model for each particular platform. Section 6 presents some experimental results with the model. Section 7 demonstrates how to use the model in parallel applications in order to improve their performance.

2. Related Work

The performance analysis of communication operations in parallel algorithms is based on models of parallel computers such as the bulk-synchronous parallel model (BSP) model [2] and the LogP model [3][4]. BSP allows for asynchronous issues of processors, network latency and bandwidth but depends on restrictive programming methodologies. The LogP model is more realistic and characterizes a parallel machine by the number of processors (P), the communication bandwidth (g), the communication delay (L), and the communication overhead (o). Many adaptations extend this model, such as LogGP [5] for long messages and LogGPC [6] to account for network contention delay. LogP is adapted for a wide area network (WAN) distributed system [7]. Another approach to modeling WAN platforms addressing the issues of network latency, bandwidth and topology proposed in [8]. Communication models of heterogeneous clusters are presented in [9,10,11].

Our model of Many-to-One communications reflects the important deviation from linearity based on our extensive experimental studies.

3. Many-to-One collective communications

The proposed performance of Many-to-One collective communications is based on our experimental observations. It describes behavior that has not been fully documented or modeled by previous research. The model describes the performance of this type of collective communication for the complete range of small, medium and large message sizes. We conducted extensive experimentation to record results that display the essential independent features of the Many-to-One operation for a typical switched Ethernet network. The experiments were repeated for platforms and different UNIX operating systems and different implementations of MPI. The model representation is verified to be independent of the operating system, the MPI implementation and the processor type by the empirical findings from our experiments.

The design of the model is based on a simple collation of the MPI send and receives from each source to a destination node (see Figure 1) that makes up the Many-to-One operation. We chose a typical network topology of switched Ethernet. This imposes a restriction referred to in the literature as '1-port full duplex' [12]. Hence, the model based on a flat-tree design is an expression (see Figure 1) of the Many-to-One communication. Any other tree, such as binomial one, will be actually transformed into a flat tree on the switched network architecture.

The experiments were conducted on many platforms with different operating systems, MPI implementations and network sizes. We found that the execution time was always linearly proportional to the message size for small messages.

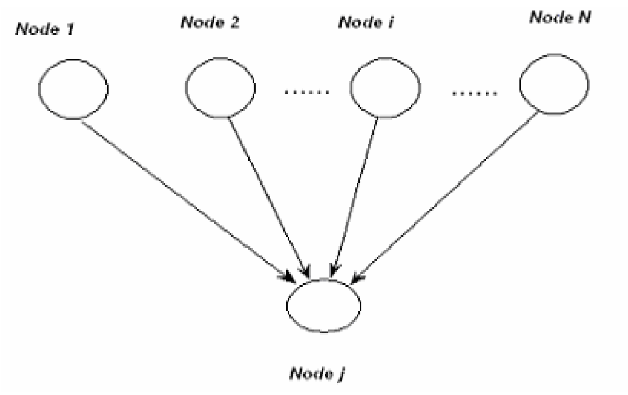

Figure 1: Many–to-One Communication

For medium-sized messages, however, we noted a marked change in behavior. Empirical readings from experiments indicated that the Many-to-One collective communication often experienced very significant delays as the message size was increased. This nonlinear behavior was manifested by very large fluctuations in execution times (see Figure 2). These extreme values were observed to occur for different networks of various size, operating systems and MPI implementation. The probability of these variations was analyzed statistically. The second part of our model is designed for medium-sized messages and describes the steep rise in execution time by a statistical estimation of the probability of the occurrence of the escalations based on experimentally obtained relative frequencies. We observe that the sharp escalations over the medium message size range form a small number of discrete levels (see Figure 2). It is interesting to note that each of the few levels is characterized by approximately *the same value,* which does not increase as the message size is increased. However, a pattern of change in relative frequencies is apparent from the empirical readings. There is a rise in the frequencies of higher discrete levels as the message size increases, while the occurrence of lower levels of escalation is seen to reduce. Moreover, beginning from some particular message size, the 100% certainty of occurrence of escalated readings were observed for medium-sized messages (see Figure 3).

These values of escalated execution times remain constant for each given platform independent on the number of processors involved in the Many-to-One communication. The frequency of occurrence of the escalation increases with the increase of the number of involved processors, but the discrete levels values remain the same.

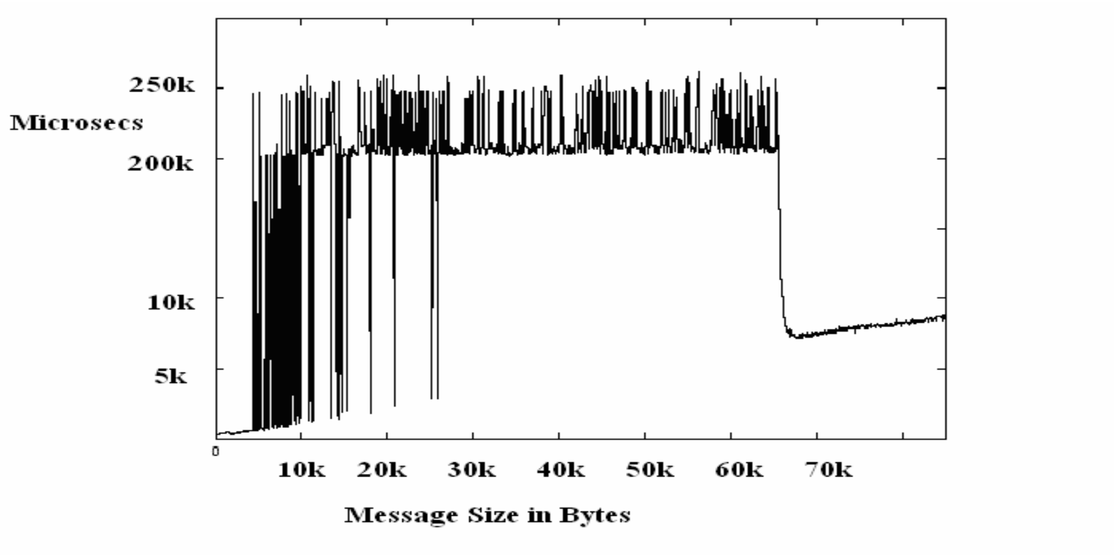

Figure 2. Performance of Many-to-One communication for medium message sizes

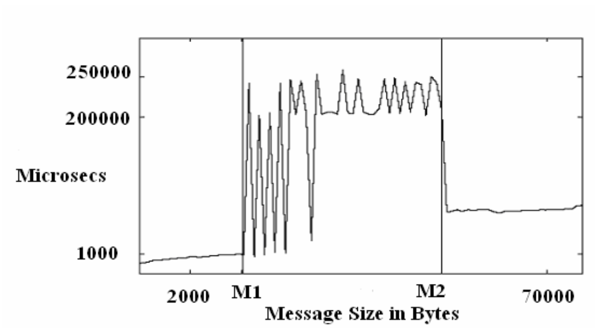

Figure 3. M_1 and M_2 message sizes for small, medium and large message regions of the Many-to-One model

For large messages, the experiments demonstrate a return to a linear response of the execution time to message size. This change in behavior occurs when the size of the message exceeds some particular threshold value (see Figure 2 and 3). Standard MPI implementation changes the sending to synchronous rather than asynchronous for large messages to allow reservation of resources for the communication operation. This provides an explanation for the sudden resumption of linearity for Many-to-One communications.

We experimented with different numbers of processors and found from the experiments that the increase of the probability of escalation was proportional to the increase in the number of processors involved in the communication. The resumption of linear readings was observed to occur at the same threshold value and is independent of the number of processors. Our model estimates for different numbers of processors with readings derived empirically for the network.

The model based on the presented observations is described in the next section in its three distinct parts for small, medium and large messages. The model represents both linear and nonlinear responses for a full range of message sizes. The first part describes the observed linear prediction with small messages. The second part describes the potential for marked escalation in execution times as message size increases. The third part is the linear performance estimation for large messages.

4. The Collective Communication Model

The Many-to-One model is an extension to our previous work [1] to design a new model for performance of all collective communications with One-to-Many and Many-to-One formations. The models are built upon the model of a single point-to-point communication. We have also proposed in [1] accurate performance models for parallel combinations of independent point-to-point communications, and for the One-to-Many communication. The Many-to-One model completes our collection of models that comprehensively estimate the performance of collective communications operations.

4.1 Point-to-point communication model

We model a single point-to-point communication operation with simple parameters found empirically for sending from the source, crossing the network and receiving at the destination. The fixed processing delay is C_i for a node *i*. The variable processing delay t_i is dependent on message size *M*, and so the communication time at node *i* as follows;

$$L_i = C_i + t_i M$$

The parameter for transmission delay found from the transmission rate of the link;

$$L_{net} = M / \beta_{ij}$$

where β_{ij} is the transmission rate. The total execution time of communicating a message of size *M* from node *i* to node *j* is expressed as follows;

$$T_{ij} = C_i + t_i M + C_j + t_j M + \frac{M}{\beta_{ij}}$$

4.2 One-to-Many model

We designed a model for One-to-Many collective communications using the parameters found with point-to-point communications. With One-to-Many communication the source node sends the same message of the size *M* to an arbitrarily number of nodes $n < N$, where *N* is the cluster size. The source node delay is much bigger than each receiving node delay due to larger amount of data to be transmitted to the receiving *n* nodes,

$$\sum_{j=1}^{n} M = n \times M \cdot$$

One-to-Many communication displays parallel communication for small messages and a serialized communication for larger messages. The port connectivity features allow this communication to be modeled as a flat tree. The model is constructed as a simple combination of parameters found with the point-to-point estimation. The execution time of the operation is estimated as follows:

$$C_0 + t_0 \times n \times M + \sum_{j=1}^{n}(C_j + t_i M + M / \beta_0) \qquad M > S$$

$$C_0 + t_0 \times n \times M + \max_{1 \le j \le n}\{C_j + t_i M + M / \beta_0\} \qquad M \le S$$

where C_0 and t_0 are the fixed and variable processing times for sending *n* messages from the source node, C_j and t_j are those for destination nodes, and *S* is the message size threshold categorizing small and large message size (described as M_2 in the next section). *S* may be different for different cluster platforms and MPI implementations.

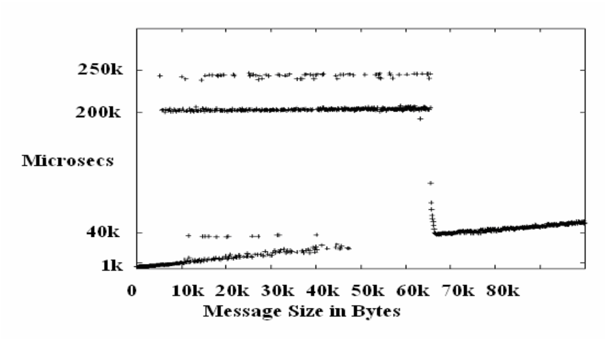

Figure 4. Escalation levels with medium messages for the Many-to-One model

4.3 Many-to-One model

The Many-to-One model is a mathematical description of the Many-to-One execution time based on our observations presented in Section 3. The proposed model has a flat tree formation. It is built from the model of a single point-to-point communication in the similar way as the One-to-Many model described in the previous sections.

The model differentiates small, medium and large messages by introducing two parameters of message size, M_1 and M_2. For small messages ($<M_1$), the execution time has a linear response to the increase of message size (see Figure 2). For medium messages, size between M_1 and M_2, the model is a mathematical estimate of the very significant degradation in performance we have found empirically. We observed three or four discrete levels of escalation, and that these levels remain constant as the message size increases. The model describes the probability of escalation to each of the levels as a function of message size and the number of processors involved in the operation. If the escalation does not occur, the linear model used for small massages accurately predicts the execution time. For large messages of the size larger than M_2, the execution time resumes a linear predictability for increasing message size.

4.3.1 Many-to-One model for small messages

The Many-to-One communication execution time for small messages increases directly in proportion to the message size. The typical network topology of switched Ethernet is modeled by a flat tree combination of sources and destination as with the One-to-Many model. The network features of port-connectivity mean that the sources act in parallel to send to the switch and then the communication is serialized to the receiver over a single link.

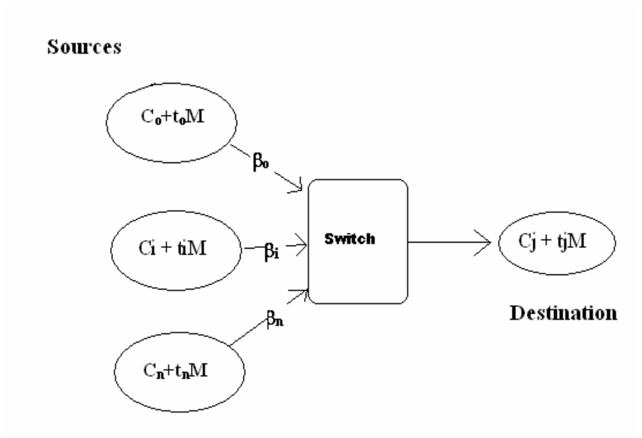

Figure 5 – Parameter values from ping-pong experiment

Figure 5 gives a graphical description of the tree formation of the model parameters for source nodes and the destination node, where parameter β_0 is the transmission time to the receiver.

Thus, for small messages, the execution time for the Many-to-One communication involving n processors ($3 \le n \le N$, where N is the cluster size) is estimated as follows:

$$T_s = (C_0 + \sum_{i=1}^{n-1} C_i) + M\,(t_0 + 1/\beta_o + \kappa_1 \sum_{i=1}^{n-1} t_i)$$

Here, κ_1 is a constant scaling parameter reflecting the contribution of a single processor into the increase of the slope of the linear function.

4.3.2 Many-to-One model for medium messages

As we further increase the message size and exceed M_1, we observe that there are some sharp escalations to a number of fixed discrete levels with very high execution times. Occurrence of the escalations is non-deterministic, and sometimes for the same operation in the same setup we observe no escalation.

In the absence of escalation, the same linear model that was used for small messages can accurately predict the execution time of the Many-to-One communication. As to the escalations, for any given combination of the cluster platform of size N and the MPI implementation, we can only predict:

- The minimum size of message M_1, beginning from which escalations can be observed.
- The maximum size of message M_2, beginning from which the escalations stop occurring.
- The minimum size of message M_c, beginning from which escalations occur with a 100% certainty $(M_1 \le M_c \le M_2)$;
- The number of levels of escalation, k;
- The execution time of the Many-to-One communication operation at each level of escalation is t_i, i={1,…, k};
- The probability of escalation to each of the levels as a function of the message size M $(M_1 \le M \le M_2)$ and the number of processors n $(3 \le n \le N)$ involved in the operation, $f_i(n,M)$, i={1,…, k}.

Parameter M_2 is a constant not depending on the number of processors n involved in the operation. Parameters M_1 and M_c depend on n. Parameter k is a constant that does not depend on n or the size of message M. Parameters $\{\ t_i\ \}_{i=1}^{k}$ are also constants that are independent of n and M.

4.3.3 Many-to-One model for large messages

Like the model for small messages in section 4.3.1, the model for large messages predicts the linear increase of the execution time with the increase of message size, based on observed behavior for messages greater than M_2 (see Figure 3). Hence, this part of the model has the same design but a different slope of linearity and greater value due to overheads. The model adds the source node overhead together with transmission time and destination processing overhead to form the estimated execution time as follows:

$$T = C_0 + t_0M + M/\beta_0 + \kappa_2 \sum_{s=1}^{n-1} (C_s + t_sM)$$

The additional parameter κ_2 accounts for the further processing requirements of large messages due to the nature of typical implementation of the standard MPI send operation. The parameter κ_2 represents the increased overhead required for synchronous sending with Many-to-One communications of large messages size $M > M_2$. The parameter also incorporates the adaptability and scalability issues of network dimension and platforms.

5. Building the model

The model is composed of a number of parameters that are determined from empirical readings achieved by a series of straightforward experiments. The parameters are defined for each particular platform characterized by its hardware, system software and MPI implementation.

- The parameters for the node's fixed C_s, C_d and variable processing time t_s, t_d are calculated by the simultaneous equations from the ping-pong latency readings between a node *s* and destination node *d*;
- The functional parameters $M_1 = M_1(n)$, $M_c = M_c(n)$ and value M_2 are determined by successive trials of the Many-to-One communication operation for all numbers of involved processors *n* ($3 \le n \le N$ where *N* is the total number of processors).
- The scalar constants κ_1, κ_2 are determined by experiment with a single Many-to-One collective communication with a message value $M < M_2$ and $M > M_2$ and are independent of processor numbers.

The parameters for the probabilistic estimation of the deviation by escalations from the linear model are found by statistical analysis of experimental readings.

- The repetition of Many-to-One operations with a constant increment finds the parameters $f_i(n,M)$ the probability of occurrence of a particular level t_i for a message size *M* with *n* nodes is found by statistical analysis. $B-spline$ coefficients are determined by regression analysis [13] by a computationally efficient interpolation method to store a representation of the escalation phenomenon.

6. Experimental results

We have experimented with a wide range of different clusters and various MPI implementations and operating systems. In this section we present the experimental results for just two platforms. The first platform has 16 heterogeneous nodes (AMD Processors, Xeon, Pentium and Celeron) running Fedora Linux, Debian Linux and SunOs operating systems. The MPI implementation is LAM 7.1.1. The second platform is the UCD Rowan HPC Cluster (a part of CosmoGrid [14]) that has thirty-two dual processors IBM x336 with 3.2GHz EM64T Intel Xeon CPUs running Fedora and Open MPI 1.0 as the MPI implementation.

The experiments determine the parameters of the model for the platforms as described in Section 5. The values of the scalar parameters of the models are given in Table 1.

Parameter	16 nodes	Rowan
C_s, C_d	*0.000081*	*0.008509*
t_s, t_d	*0.00426*	*0.0031704*
β_0	*0.000008*	*0.0000004*
M_2	*65536*	*65536*
κ_1	*0.33*	*0.146*
κ_2	*1.95*	*1.74*

Table 1. Scalar parameter values

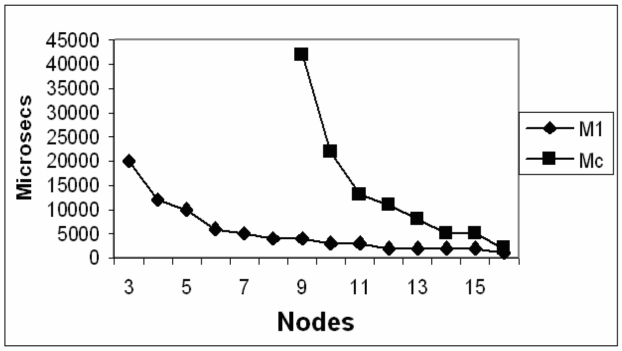

Figure 6. Parameter values M_1 and M_c for the sixteen node cluster

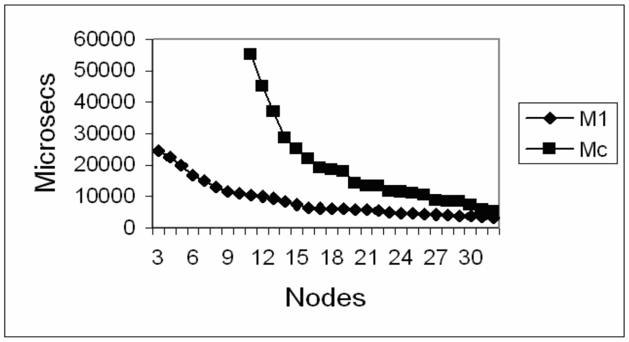

Figure 7. Parameter values M_1 and M_c for the thirty-two node Rowan cluster

Figures 6 and 7 presents the functional parameters M_1 and M_c for the 16-node cluster and the thirty-two node Rowan platforms. The four levels of escalation were found with experiments for the Rowan cluster. The levels of escalation $\{ t_i \}_{i=1}^{k}$ are summarized in Table 2. Figures from 8 to 11 show the functional parameters $f_i(n,M)$ for Rowan.

Levels of Escalation	*Value in Microseconds*
t_1	*40,000*
t_2	*200,000*
t_3	*250,000*
t_4	*600,000*

Table 2. Levels of escalation with Many-to-One collective communications for Rowan cluster

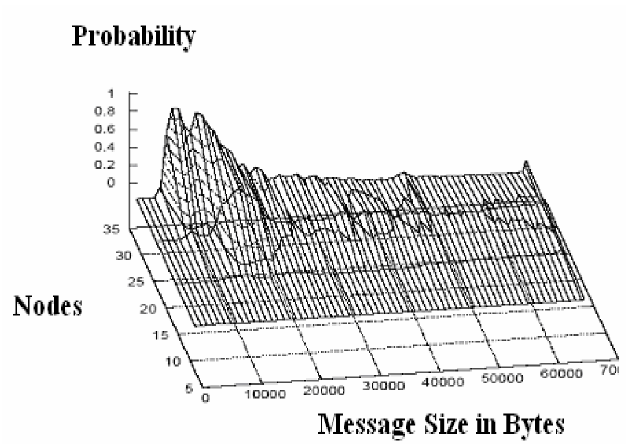

Figure 8. Surface of probability escalation to 40k microseconds

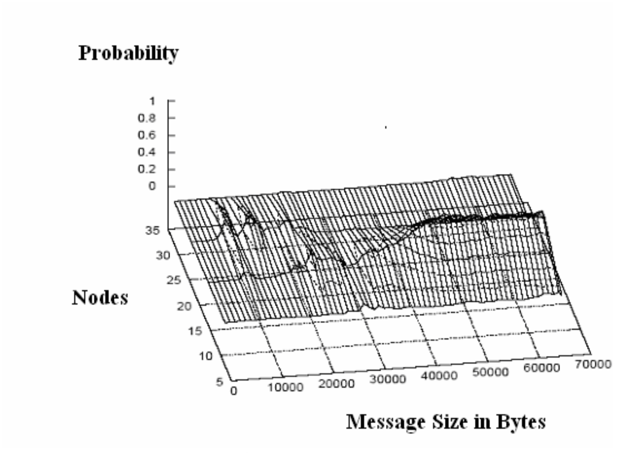

Figure 9. Surface of probability escalation to 200k microseconds

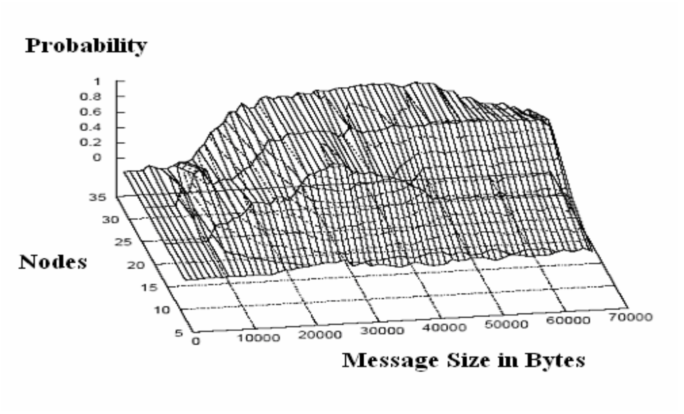

Figure 10. Surface of probability of escalation to 250k microseconds

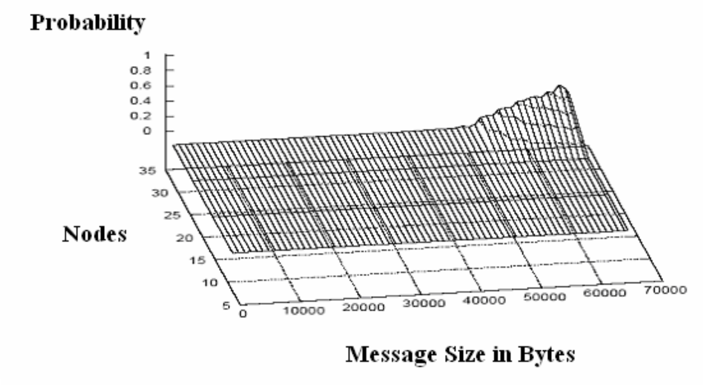

Figure 11. Surface of probability of escalation to 600k microseconds

7. Application

Multi-spectral satellite image sets are produced by satellites at *512 x 512* resolution [15]. We demonstrate the effectiveness of the model with a typical real-time satellite imaging application using parallel processing. The host processor receives a sequence of satellite raw data images, each of *512 x 512* bytes (one byte per pixel). Each image is divided into partitions for parallel processing by a cluster of workstations. The partitions are processed in parallel and then returned to the host processor with the MPI_Gather operation for visualization. This application as described will face one serious performance problem. The point is that when the number of processors is increased the partition sizes will decrease. If for a small number of processors the partition size fits into the area of large messages, then an increase in number of processors will move it into the area of medium messages $M_1 \le M \le M_2$. This will result in the significant increase in execution times of the single MPI_Gather operation.

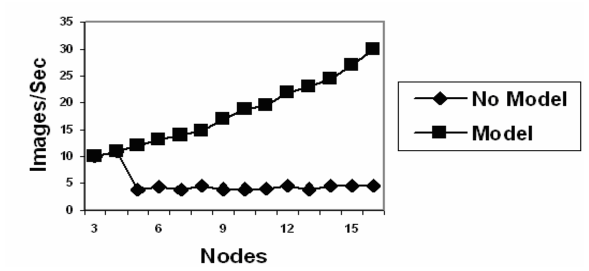

Figure 12. Frames per second for traditional application and compared to redesigned applications

We redesign this application to address this issue as follows. We replace the single MPI_Gather gathering medium-sized messages by an equivalent sequence of MPI_Gather operations, gathering messages with a size that fits the range of small messages. Thus, each medium-sized partition will be communicated as a sequence of small-sized sub-partitions. More precisely we calculate the number of sub-partitions m of a partition of the medium size M so that

$$\frac{M}{m} \le M_1 \text{ and } \frac{M}{m-1} > M_1 .$$

We compared the performance of the original and redesigned applications with our experiments with 16 workstations. Figure 12 shows the number of images per second that can be visualized on the host workstation by the original and redesigned applications against the number of workstations involved in parallel processing.

8. Conclusion

We present a model to estimate Many-to-One collective communications both small and large message sizes. The model is a representation of both linear and nonlinear performance with the regression analysis of empirical readings. The model is applicable to represent clusters of different sizes for standard MPI implementations. Our aspirations for further work are to extend the model represent performance of collective communications on wide-area networks. We demonstrate with experiments that the model is a computationally simple and accurate representation for Many-to-One collective communications. We demonstrated the importance of model for the efficient design of applications in parallel computing.

The work was supported by Science Foundation Ireland.

9. References

[1] A. Lastovetsky and I. Mkwawa and M. OFlynn, "An Accurate Communication Model for a Heterogeneous Cluster based on a Switch-Enabled Ethernet Network", *The 12th International Conference on Parallel and Distributed Systems (ICPADS 2006)*, 2006.

[2] L.G. Valiant, "A Bridging Model for Parallel Computation", *Communications of the ACM*, Vol. 33, No. 8, August 1990.

[3] D. Culler and R. Karp and D. Patterson and A. Sahay and K. E. Schauser and E. Santos, R. Subramonian and T. von Eicken, "LogP: Towards a Realistic Model of Parallel Computation*", Proceedings of the Fourth ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming,* San Diego, CA, May, 1993.

[4] A.L. Lastovestsky, "*Parallel Computing on Heterogeneous Networks*, Wiley Series on Parallel and Distributed Computing, Wiley, 2003.

[5] A. Alexandrov, M. F. Ionescu, K. E. Schauser, C. Scheiman, "LogGP: Incorporating Long Messages into the LogP Model", *Journal of Parallel and Distributed Computing*, Vol. 44, Number 1, July 1997, pp 71-79.

[6] C. Moritz, M. Frank, "Logpc: Modeling network contention in message passing programs*", IEEE Transactions on Parallel and Distributed Systems, April 2001*, 12(4): 1-100.

[7] T. Kielmann, R. Hofman, H. Bal, A. Plaat, R. Bhoedjang, "MagPIe: MPI's Collective Communication Operations for Clustered Wide Area Systems", *Seventh ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, Atlanta, GA, ACM Press. (1999) pp. 131-140.

[8] L. Marchal and Y. Yang and H. Casanova and Y. Robert, "A Realistic Network/Application Model for Scheduling Divisible Loads on Large-Scale Platforms", *19th IEEE International Parallel and Distributed Processing Symposium (IPDPS'05)*, 2005, pp 48b.

[9] O. Beaumont and A. Legrand and Y. Robert, "The Master-Slave Paradigm with Heterogeneous Processors", *3rd IEEE International Conference on Cluster Computing (CLUSTER'01),* 2001, pp 419.

[10] J. Bosque and L. Perez, "HLogGP: A New Parallel Computational Model for Heterogeneous Clusters*", 4th IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid 2004)*, 2004, pp 403-410

[11] S. Vadhiyar and G. Fagg and J. Dongarra, "Towards an Accurate Model for Collective Communications", *The International Journal of High Performance Computing Applications*, 2004, pp 159.

[12] L. A. Stefenel, "Modeling Network Contention Effects on All-to-All Operations", *Cluster 2006, Barcelona*, September 2006.

[13] T. Blue and M. Unser, "Minimum support interpolators with optimum approximation properties", *Int. Conf. Image Processing*, Santa Barbara, CA, 2005.

[14] The Cosmogrid Project 2004, http:// www.cosmogrid.ie/.

[15] M Fluery, R Self, A Downton, "Multi-spectral Satellite Image Processing on a Platform FPGA Engine", *Military and Aeronautics Logic Devices (MAPLD'05)*, Washington D.C., 2005.