Communication Models for Resource Constrained Hierarchical Ethernet Networks

Jun Zhu[1], Alexey Lastovetsky[2], Shoukat Ali[3], and Rolf Riesen[3]

[1] Technical University of Eindhoven, Netherlands
[2] University College Dublin, Ireland
[3] Dublin Research Laboratory, IBM, Ireland

Abstract. Communication time prediction is critical for parallel application performance tuning, especially for the rapidly growing field of data-intensive applications. However, making such predictions accurately is non-trivial when contention exists on different components in hierarchical networks. In this paper, we derive an 'asymmetric network property' on TCP layer for concurrent bidirectional communications in a commercial off-the-shelf (COTS) cluster, and develop a communication model as the first effort to characterize the communication times on hierarchical Ethernet networks with contentions on both NIC and backbone cable levels. In particular, we show that if the asymmetric network property was excluded from the model, the communication time predictions will be significantly less accurate than those made by using the asymmetric network property. Furthermore, our observation of the performance degradation caused by the asymmetric network property suggests that some part of the software stack below TCP layer in COTS clusters needs targeted tuning, which has not yet attracted any attention in literature.

1 Introduction

Computing clusters have been the primary commodity platforms for running parallel applications. To build a cost-effective yet powerful cluster environment, high speed networks are widely used to interconnect off-the-shelf computers. For application performance analysis on such clusters, an accurate time prediction for data sets transferred over the communication media is typically required [4]. To that end, several communication models have been proposed, e.g., Hockney [8] and LogP [6]. They have been widely used to analyze the timing behaviors for point-to-point communications on parallel computers. However, these models simply see the network as a black-box, and can capture neither the communication *hierarchy* nor the network *resource sharing* that typically is present in modern large scale systems. Both of these factors, network *hierarchy* and *resource sharing*, make communication time prediction non-trivial and challenging for high performance clusters.

On the other hand, such predictions are needed more now than ever because of the increasing importance of data-intensive applications [7][10] that devote a significant amount of their total execution time in parallel processing to I/O or

D. an Mey et al. (Eds.): Euro-Par 2013 Workshops, LNCS 8374, pp. 259–269, 2014.

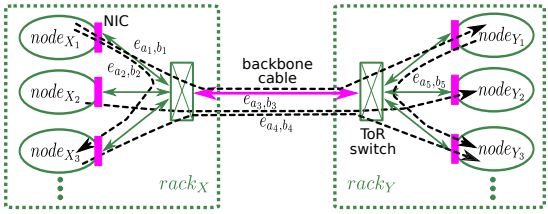


Fig. 1. A tree topology platform: two star-configured racks connected via the backbone cable. The dashed arrows denote one example application with five logical communication links: e_{a_1,b_1}– e_{a_5,b_5}. The processes on each logical link are not explicitly labeled for clarity in the graph.

network communication, instead of computation. A good usable performance analysis of such data-intensive applications requires that the communication model reflect the network properties accurately on state-of-the-art network topologies and technologies.

An example Ethernet cluster, consisting of two racks, is illustrated in Figure 1. It has a scalable tree (star bus) topology, i.e., two star-configured intra-rack segments are connected using a linear backbone cable. The computer nodes in the same rack are connected to a *top-of-rack* (*ToR*) switch via network interface cards (NICs), and each node has a dedicated bandwidth on the intra-rack point-to-point connection. Different ToR switches are connected together to form a hierarchical network. We use this cluster as our testbed system to verify network properties and the proposed communication model. In this testbed, contentions may occur at different levels of the communication infrastructure:

- Network interface cards (*NICs*). In multi-core processors, multi-socket nodes, or a combination thereof, each NIC may be shared by multiple cores concurrently. That is, the number of NICs on each node is generally less than the number of cores and NIC-level sharing exists.
- Backbone cable. Several inter-rack communication operations may be aggregated on the backbone that runs between racks to effectively utilize the high bandwidth available.

For instance, in Figure 1, logical communication links e_{a_1,b_1} and e_{a_2,b_2} share the same NIC on $node_{X_1}$, and e_{a_1,b_1}, e_{a_3,b_3} and e_{a_4,b_4} share the inter-rack backbone instead. When resource contention happens on a *hierarchical* network, it makes communication times prediction more difficult.

Contributions. We derive network properties on parametrized network topology from simultaneous point-to-point MPI operations, and discover the *asymmetric network property* on TCP layer for concurrent bidirectional communications in a COTS Ethernet cluster. To the best of our knowledge, it has not been previously reported in literature, and our work is the first effort to characterize the effect of concurrent communications in *resource-constrained hierarchical* networks. In particular, we show that if the asymmetric network

property is excluded from the model, the communication time predictions become significantly less accurate than those made by using the asymmetric network property.

The remainder of this paper is structured as follows. Section 2 discusses some related work. Our MPI micro-benchmark and platform are introduced in Section 3. Section 4 introduces the notations used in this paper. Section 5 introduces some network properties derived from benchmarking. We propose our communication model in Section 6, and present the corresponding experiments in Section 7. Finally, Section 8 concludes the paper.

2 Related Work

Many models have been proposed in the parallel and distributed computing literature to characterize the communication times for parallel computers. In the *Hockney model* [8], the point-to-point communication time to send a message of size m is captured as $\alpha + \beta * m$, where α is the latency for each message and β is the inverse of the network bandwidth. Culler *et al.* [6] describe *LogP* communication model for small messages. The model is aware of the finite network capacity, which is the upper bound on messages in parallel transmission on the network. Hoefler *et al.* [9] further investigate how this model can be modified to capture the timing behaviors for small messages on InfiniBand networks. In [3], *LogGP* is developed, as an extension of the LogP model, to better fit large messages. In [11], a heterogeneous model *LMO* is proposed, which separates the variable contributions of the processors and network in communication timing analysis. However, all of the above work assumes that there is no contention on the network. When multiple communication transactions happen concurrently on the same network resource, the above models assume that the available bandwidth is even shared by all concurrent communications. Let us call this the *symmetric network property*. We will show that this property does not exist for complicated hierarchical networks.

In a recent work [12], a contention-aware communication timing model is proposed for multi-core clusters using an InfiniBand network. This work analyzes the dynamic network contention, and derives the so-called *penalty coefficients* to characterize the bandwidth distribution at the NIC level. This work is similar to ours in the sense that it does not recognize the symmetric network property; instead it explicitly calculates how the total available bandwidth will be divided among the contending communications. This model focuses on a flat star network, in which all computer nodes are connected to a single switch with dedicated bandwidth. However, modern large-scale systems usually have hierarchical network topology. This is the part that had been lacking, and this is what we address in our paper.

3 Micro-benchmark and Platforms

3.1 Micro-benchmark

We designed a point-to-point MPI micro-benchmark to measure concurrent communication times for use in our testbed. In each iteration, a message with a user-specified size is initialized to remove the potential memory or network cache effects. Each time a message is sent from the sender to receiver, a non-blocking receive operation is pre-posted and the receiver records the time spent on message transmission in `tArray` for statistics. For each sender to receiver communication, we repeat the transmission at most `maxIter` times, where `maxIter` is some large number (set to 2000 in our experiments). Specifically, we stop if the width of the 95% confidence interval becomes 2%, or less, of the average measured time. In general, our benchmark can be set up to have any given number of pairs of sender and receiver processes for simultaneous MPI communication operations.

We use OpenMPI 1.5.4 [2] as the MPI implementation. Each MPI process can be bound to any computing resource (core or node) as specified in a given communication pattern, with the support of *processor affinity* in OpenMPI. We only use large message sizes $\geqslant 10MB$ in benchmarking, which suit data-intensive problems. Compared with communication delay, the message latency α (also called propagation delay) is negligible [12].

3.2 Platform Specifications

There are up to 15 nodes settled in each rack of our experimental platform. Each node is a dual-socket six-core (Intel Xeon X5670 6C@2.93GHz) server with an Intel 1Gb NIC card, operated with Red Hat Enterprise Linux 5.5 x86-64. The Ethernet switch is IBM BNT Rack Switch G8264, which supports over 10Gb Ethernet and is interoperable with clients using 1Gb connectivity as well. The theoretical communication bandwidth on different network resources is 1Gbps and 10Gbps for NICs and the optical backbone respectively.

The NIC settings in Linux on each node are tuned to run for Gigabit speed [5]. In our hybrid 1/10 GbE cluster environment, the network round-trip time (RTT) between two nodes on different racks is 450 μs. The socket buffer size is at least RTT * bandwidth. We set the TCP socket buffer size in OpenMPI to 3MB (MCA parameters `btl_tcp_sndbuf` and `btl_tcp_rcvbuf`).

4 Preliminaries

Here, we introduce the notations on application and platform, which will be used to formalize network properties and communication models in the following sections. Let $|\cdot|$ denote the cardinality of a set or vector, and $(\cdot)^{-1}$ the inverse of a value. The index of a vector starts at 0, and the *i-th* element of a vector $\boldsymbol{V}$ is denoted as $\boldsymbol{V}[i]$.

4.1 Application

An application is a collection of concurrent point-to-point MPI operations, denoted as a tuple (P, E). A finite set P contains an even number of processes, which are connected in pairs via a finite set $E \subseteq P^2$ of edges, with $|P| = 2 * |E|$. Each edge $e_{a,b} \in E$ denotes a logical communication link (*dependency*) between a process pair: a sender p_a and a receiver p_b.

4.2 Platform

The network infrastructure of our experimental cluster has a tree topology, as illustrated in Figure 1. In general, the cluster consists of a set R of racks. The set N of computing nodes is the union of all nodes in individual racks. There is one NIC configured for each node and all nodes in the same rack communicate via the ToR switch. A full-duplex communication network is employed in our work. We denote the inverse bandwidth of the NIC, electrical backbone, and optical backbone as β_N, β_E, and β_O, respectively.

4.3 Mapping: Application to Platform

The mapping process binds each process in MPI applications to a computing node in our platform. The binding function is defined as $\mathcal{B}^N : P \rightarrow N$, which associates every process $p \in P$ to a *node* $\in N$ to which it is bound. Similarly, how processes are bound to racks is defined $\mathcal{B}^R : P \rightarrow R$. For instance, in Figure 1, $\mathcal{B}^N(p_{a_5}) = node_{Y_1}$ and $\mathcal{B}^R(p_{a_5}) = \text{rack}_Y$. In our work, we only map up to one process to each core to eliminate the unnecessary context switching overhead.

4.4 Network Contention

When multiple processes are bound to one (multi-core) node or rack, several simultaneous logical links may share the same NIC or inter-rack backbone. In full-duplex network, we distinguish resource contention on incoming links from that on outgoing links. The set of incoming logical links that are bound to the same resource $x \in N \cup R$, are denoted as $\mathcal{E}^+(x)$. The degree of this set $|\mathcal{E}^+(x)|$ is simply denoted as $\delta^+(x)$, where $\delta^+(\cdot)$ is the indegree function for a resource defined as $\delta^+ : N \cup R \rightarrow \mathbb{N}_0$. Similarly, $\mathcal{E}^-(\cdot)$ and $\delta^-(\cdot)$ are defined for outgoing logical links of a resource. For instance, $\mathcal{E}^-(node_{X_3}) = \{e_{a_4,b_4}\}$ and $\delta^-(\text{rack}_X) = 3$ in Figure 1.

For simultaneously bidirectional communication (Section 5.2), we also distinguish logical links with resource sharing at the *same* direction (*with-flow*) to logical links on the *reverse* direction (*contra-flow*).

Congestion Factors. To detect the communication *bottleneck* in a *hierarchical* network, we associate with $e_{a,b}$ a vector $\boldsymbol{S}_{a,b}$ of sets of logical links, defined as follows

$$\boldsymbol{S}_{a,b} =< \mathcal{E}^-(\mathcal{B}^R(a)), \mathcal{E}^-(\mathcal{B}^N(a)), \mathcal{E}^+(\mathcal{B}^N(b)) > \quad (1)$$

where $\boldsymbol{S}_{a,b}$ includes 3 sets of ***with-flow*** logical links of $e_{a,b}$ on the shared resource. For instance, the link e_{a_1,b_1} in Figure 1 has

$$\boldsymbol{S}_{a_1,b_1} =< \{e_{a_1,b_1}, e_{a_3,b_3}, e_{a_4,b_4}\}, \{e_{a_1,b_1}, e_{a_2,b_2}\}, \{e_{a_1,b_1}\} >$$

To detect the congestion bottleneck of a logical link on network resources, a congestion factor $k_{a,b}$ is defined as follows

$$k_{a,b} = \max\{k \mid k = \boldsymbol{\beta}[i] \cdot |\boldsymbol{S}_{a,b}[i]|, \forall i \in [\,0, |\boldsymbol{S}_{a,b}|\,)\} \tag{2}$$

where $\boldsymbol{\beta}$ is a vector with platform-dependent inverse bandwidth values, and $\boldsymbol{\beta}[i]$ the *i-th* inverse bandwidth of the network resource on which the set $\boldsymbol{S}_{a,b}[i]$ of logical links are sharing. For instance, when the cluster in Figure 1 has an optical backbone cable, $\boldsymbol{\beta} =< \beta_O, \beta_N, \beta_N >$ and $k_{a_1,b_1} = \max\{3\beta_O, 2\beta_N, \beta_N\}$. That is, the congestion factor is configuration aware in a hybrid network.

Similarly, vector $\bar{\boldsymbol{S}}_{a,b}$ of sets of ***contra-flow*** logical links of $e_{a,b}$ and the *reverse* congestion factor $\bar{k}_{a,b}$ are defined

$$\bar{\boldsymbol{S}}_{a,b} =< \mathcal{E}^+(\mathcal{B}^R(a)), \mathcal{E}^+(\mathcal{B}^N(a)), \mathcal{E}^-(\mathcal{B}^N(b)) > \tag{3}$$

$$\bar{k}_{a,b} = \max\{k \mid k = \boldsymbol{\beta}[i] \cdot |\bar{\boldsymbol{S}}_{a,b}[i]|, \forall i \in [\,0, |\bar{\boldsymbol{S}}_{a,b}|\,)\} \tag{4}$$

5 Network Properties

We derive some network properties from MPI benchmarking on a resource constrained network platform. In this section, we characterize the inverse bandwidth $\beta_{a,b}$ of each logical link $e_{a,b}$ in a particular communication pattern, which is time independent. However, when some communication operations finish earlier, the specified communication pattern may vary at different time instance t.

5.1 Unidirectional Communication: Fairness Property

Given an application with a number $|E|$ of point-to-point MPI operations, we design the following unidirectional experiments to inspect the arbitration fairness on different levels of the network hierarchy:

A. **Intra-rack communication** – all sender processes are mapped onto one *node*, while the matching receiver processes are mapped onto another *node* in the same rack.
B. **Inter-rack communication** – all sender processes are mapped onto different *nodes* in rack_X, while the matching receiver processes are mapped onto different *nodes* in rack_Y.

We observe that when $|E|$ increases, the average bandwidth for logical links on the shared NIC (experiment A) decreases, and that the bandwidth is fairly distributed over all links. The same is for the optical fiber backbone (experiment

B) when $|E| > 10$. When $|E| <= 10$, the 10-Gb optical fiber is not saturated and therefore the average bandwidth is almost constant (940 Mbps), giving a measured maximum aggregate bandwidth $\beta_O^{-1} = 9.4$Gbps. For experiment B using electrical copper backbone, the results are similar to those for a NIC. However, the bandwidth distribution in Experiment B does not really depend on copper versus optical; it is the bandwidth of physical links in the hierarchical network that matters.

Mathematically, the inverse bandwidth $\beta_{a,b}$, based on a *fairness property* of each logical link $e_{a,b}$ can be captured as

$$\beta_{a,b} = \begin{cases} \beta \cdot |E|, \text{ if } \beta = \beta_O \text{ and } |E| > 10 \text{ or } \beta = \beta_E \\ \beta_E, \text{ if } \beta = \beta_O \text{ and } |E| \leqslant 10 \end{cases} \tag{5}$$

where β is the inverse bandwidth of the physical link on which $e_{a,b}$ is located.

5.2 Bidirectional Communication: Asymmetric Property

In a full-duplex network, the network resources might be shared by multiple communication logical links in both directions simultaneously. To study bidirectional communication on shared network resources, we swap the mapping policy for some of the sender and receiver processes in the experiments in Section 5.1. When the number of *incoming* $\delta^+(\cdot)$ and *outgoing* $\delta^-(\cdot)$ logical links vary on the shared node or rack, The inverse bandwidth $\beta_{a,b}$ of each logical link $e_{a,b}$ can be captured

$$\beta_{a,b} = \begin{cases} \beta \cdot \delta_{max}(\cdot), \text{ if } \beta = \beta_O \text{ and } \delta_{max}(\cdot) > 10 \text{ or } \beta = \beta_E \\ \beta_E, \text{ if } \beta = \beta_O \text{ and } \delta_{max}(\cdot) \leqslant 10 \end{cases} \tag{6}$$

where $\delta_{max}(\cdot) = \max\big(\delta^+(\cdot), \delta^-(\cdot)\big)$. The results clearly show that the total duplex bandwidth may not be achieved, when $\delta^+(\cdot)$ and $\delta^-(\cdot)$ do not match (are *asymmetric*) on the shared resource. For instance, when $\delta^+(\cdot) = 12$ and $\delta^-(\cdot) = 1$ in (a), the total bidirectional bandwidth is $\frac{13}{12} \cdot 940$Mbps, instead of $2 \cdot 940$Mbps (according to a fair dynamic bandwidth allocation in full duplex-mode). For instance, when $\delta^+(\cdot) = 2$ and $\delta^-(\cdot) = 1$ in (a), the two incoming and one outgoing logical links all have bandwidth 470Mbps, instead of 470Mbps, 470Mbps, and 940Mbps respectively (according to a fair dynamic bandwidth allocation in full duplex-mode). We have validated this property using a TCP network testing tool Iperf [1] with the same experimental settings on TCP layer. To the best of our knowledge, this bidirectional *asymmetric property* on full-duplex communication network has not been reported in the literature yet.

6 Resource Constrained Communication Model

Here, we present our communication model, in which the inverse bandwidth for logical links can be derived. Let $\bar{E}$ be the set that stores the links for which $\beta_{a,b}$ has been calculated, E the set of links not yet calculated. $PQ(E)$ is a priority

queue based on elements $\forall e_{a,b} \in E$, which is ranked by the descending-order of a multi-key

$$\mathcal{K}_{PQ(E)} =< \max(k_{a,b}, \bar{k}_{a,b}),\ k_{a,b},\ \bar{k}_{a,b} > \tag{7}$$

The most congested logical link in a hierarchical network is the one with the highest value of $\mathcal{K}_{PQ(E)}$, which depends both on network capacity and the number of logical links on the sharing network resources.

Algorithm 1: Inverse bandwidth for logical links

Output: Inverse bandwidth $\beta_{a,b}$, $\forall e_{a,b} \in E$

/* *Initialization: set* $\bar{E}$ *to store calculated links* */
$\bar{E} := \emptyset$
while $E \neq \emptyset$ **do**
  /* *To retrieve the most congested link in* $PQ(E)$ */
  $e_{a,b} := PQ(E)$.`pop()`
  $\beta_{a,b} := 0$
  if $k_{a,b} \geqslant \bar{k}_{a,b}$ **then**
    foreach $i \in [0, |\boldsymbol{S}_{a,b}|]$ **do**
      if $\boldsymbol{\beta}[i] \cdot |\boldsymbol{S}_{a,b}[i]| == k_{a,b}$ **then**
        $\beta_{a,b} := \max(\beta_{a,b}$, `inverseBandwidth(`$i$`)`$)$
  else
    foreach $i \in [0, |\bar{\boldsymbol{S}}_{a,b}|]$ **do**
      if $\boldsymbol{\beta}[i] \cdot |\bar{\boldsymbol{S}}_{a,b}[i]| == \bar{k}_{a,b}$ **then**
        $E' = \bar{\boldsymbol{S}}_{a,b}[i] \cap \bar{E}$
        /* *If contra-flow links saturate the physical link* */
        if $\sum\{\beta^{-1}_{x,y} \mid \forall e_{x,y} \in E'\} == \boldsymbol{\beta}^{-1}[i]$ **then**
          $\beta_{a,b} = \min\{\beta_{x,y} \mid \forall e_{x,y} \in E'\}$
        else
          $\bar{\boldsymbol{S}}_{a,b}[i] := \emptyset$
          $PQ(E)$.`insert(`$e_{a,b}$`)`
  /* *To update* E *and* $\bar{E}$*, once* $k_{a,b}$ *is calculated* */
  if $\beta_{a,b} \neq 0$ **then**
    $\bar{E} := E \setminus \{e_{a,b}\}$
    $\bar{E} := \bar{E} \cup \{e_{a,b}\}$
where
Function `inverseBandwidth(`i`)`
  $E' = \boldsymbol{S}_{a,b}[i] \cap \bar{E}$, $E'' = \boldsymbol{S}_{a,b}[i] \cap E$
  $\hat{\beta} = \boldsymbol{\beta}[i]$
  /* *The used bandwidth on the congested physical link* */
  $k' = \sum\{\hat{\beta}^{-1}_{x,y} \mid \forall e_{x,y} \in E'\}$
  /* *To calculate link bandwidth based on fairness property* */
  return $\beta_{a,b} := \max\left(\beta_E, \frac{|E''|}{\hat{\beta}^{-1} - k'}\right)$

Based on the derived network properties and heuristics from benchmarking, we propose Algorithm 1 to calculate the inverse bandwidth for logical Links with simultaneous MPI communications. The analysis flow works iteratively in a bottleneck driven manner. In each iteration, the most congested logical link in

E is proposed to be analyzed (Line 5), and the bandwidth of this logical link is calculated differently when the network congestion is caused by either with-flow or contra-flow traffic:

- When congestion happens on with-flow traffic (Line 7), the bandwidth is calculated based on the fairness properties on the with-flow bottle network resource (Line 9-10).
- Otherwise (Line 11), the bandwidth is calculated based on the asymmetric properties on the contra-flow bottle network resource (Line 13-17). Heuristically, when contra-flow congestion happens on a non-saturated physical link, the contra-flow congestion on the logical link is disabled and the logical link is sent back to the queue to be re-analyzed (Line 19-20).

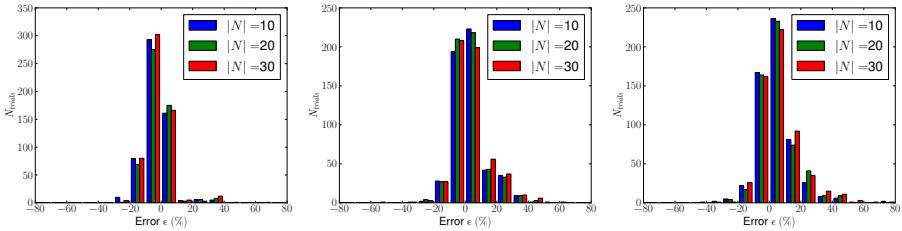
While E is not empty, the algorithm explores the links iteratively until all the logical links are analyzed. In the worst case, Algorithm 1 terminates in at most $2 \cdot |E|$ iterations. Furthermore, our communication model could be extended to more complex topologies, such as 2D mesh, in which similar network contention happens in different regions in the network.

7 Experiments and Results

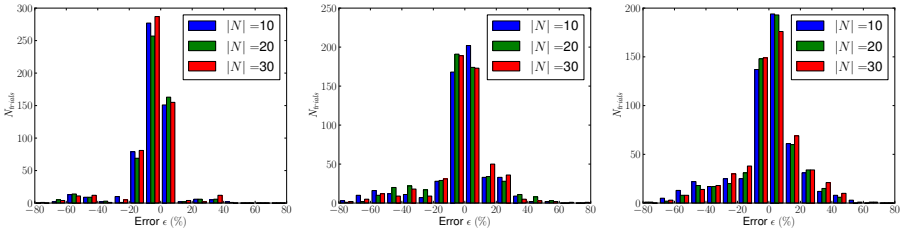
We conducted our experiments on two racks connected via an optical backbone. Each time the same number of nodes are configured in both racks, with a total number of nodes $|N|$ up to 30. To construct one test, we in turn consider each one of the nodes on both racks. For each such node, $node_{\text{src}}$, we randomly select a different node, $node_{\text{dst}}$, from the set N. We then include the directed point to point communication $node_{\text{src}} \rightarrow node_{\text{dst}}$ (as one instance of process pair in our benchmark) in the test with a 50% probability. For each $node_{\text{src}}$, we perform this matching process d times. It ensures that we get random communication patterns in the test, and that our results do not depend on a "lucky" selection of communication patterns. To further ensure that the goodness of our reported results is not a result of biased selection of input, we consider an experiment completed only when enough tests have been performed to give us a certain level of confidence in the average value of ϵ.

We have designed 9 experiments, each with a different set of values for parameters $|N|$ and d, as illustrated in Figure 2. In each experiment, the number of communication links is $N_{\text{trials}} > 500$. A total of 354 randomly generated communication patterns are tested. These communication patterns are non-trivial, with the maximum number of logic links $|E|$ in one pattern up to 57, and the maximum indegree $\delta^+(\cdot)$ or outdegree $\delta^-(\cdot)$ (i.e., the number of concurrent links) on the congested network resource up to 6 for NICs and 10 for the optical backbone.

The distribution of the estimation error ϵ on each cluster of experiments is illustrated in Figure 2. For communication times prediction based on our proposed model (the first row in Figure 2), when d varies from 1 to 3 in experiment settings, there are 83.2%, 77.3%, and 72.1% of communication links respectively, which fall within the margin of error $|\epsilon| \leqslant 10\%$. On the contrary, when the asymmetric property is not considered (the second row in Figure 2), i.e., only with



(a) Proposed model: $d = 1$ (b) Proposed model: $d = 2$ (c) Proposed model: $d = 3$



(d) Pure fairness: $d = 1$ (e) Pure fairness: $d = 2$ (f) Pure fairness: $d = 3$

Fig. 2. The histogram of errors on communication times prediction

fairness property, the percentage of these links fall to 66.8%, 58.0%, and 68.8%. The prediction error with pure fairness property can be as worse as −80%, which means the predicted times are 5 times lower than the measured ones. From the experimental results, we see that our model is quite accurate from the averaged value for error $|\epsilon|$. The largest average error occurs for experiment 9, being approximately 9.5% of the measured value with less than 0.2% imprecision.

8 Conclusions and Future Work

In this paper, we derive an 'asymmetric network property' on TCP layer for concurrent bidirectional communications on Ethernet clusters, and develop a communication model to characterize the communication times accordingly. We conduct statistically rigorous experiments to show that our model can be used to predict the communication times for simultaneous MPI operations on resource constrained networks quite effectively. In particular, we show that if the asymmetric network property is excluded from the model, the accuracy of the communication time prediction drops significantly.

As the future work, we plan to generalize our model for more complex network topologies. We would also like to investigate how the asymmetric network property can be tuned below TCP layer in Ethernet networks.

Acknowledgement. The authors would like to thank Kiril Dichev and Dr. Konstantinos Katrinis for useful discussions. The research in this paper was supported by IRCSET(Irish Research Council for Science, Engineering and Technology) and IBM, grant number IRCSET-IBM-2010-06.

References

1. Iperf site, `http://sourceforge.net/projects/iperf/` (retrieved January 24, 2012)
2. OpenMPI site, `http://www.open-mpi.org` (retrieved January 24, 2012)
3. Alexandrov, A., Ionescu, M.F., Schauser, K.E., Scheiman, C.: LogGP: incorporating long messages into the LogP model – one step closer towards a realistic model for parallel computation. In: P. of. the Seventh Annual ACM Symposium on Parallel Algorithms and Architectures, SPAA 1995, pp. 95–105 (1995)
4. Bell, G., Gray, J., Szalay, A.: Petascale computational systems. Computer 39, 110–112 (2006)
5. Leitao, B.H.: Tuning 10Gb network cards on Linux. In: Ottawa Linux Symposium, pp. 169–184 (2009)
6. Culler, D., Karp, R., Patterson, D., Sahay, A., Schauser, K.E., Santos, E., Subramonian, R., von Eicken, T.: LogP: towards a realistic model of parallel computation. SIGPLAN Not. 28, 1–12 (1993)
7. Gokhale, M., Cohen, J., Yoo, A., Miller, W.M., Jacob, A., Ulmer, C., Pearce, R.: Hardware technologies for high-performance data-intensive computing. Computer 41(4), 60–68 (2008)
8. Hockney, R.W.: The communication challenge for MPP: Intel Paragon and Meiko CS-2. Parallel Comput. 20, 389–398 (1994)
9. Hoefler, T., Mehlan, T., Mietke, F., Rehm, W.: LogfP - A Model for small Messages in InfiniBand. In: P. of. the 20th IEEE Int'l Parallel & Distributed Processing Symposium (IPDPS), PMEO-PDS 2006 Workshop (April 2006)
10. Johnston, W.E.: High-speed, wide area, data intensive computing: A ten year retrospective. In: P. of. the 7th IEEE Int'l Symposium on High Performance Distributed Computing, HPDC 1998, pp. 280–291 (1998)
11. Lastovetsky, A., Rychkov, V., O'Flynn, M.: Accurate heterogeneous communication models and a software tool for their efficient estimation. Int'l J. of High Performance Computing Applications 24, 34–48 (2010)
12. Martinasso, M., Méhaut, J.-F.: A contention-aware performance model for HPC-based networks: A case study of the InfiniBand network. In: Jeannot, E., Namyst, R., Roman, J. (eds.) Euro-Par 2011, Part I. LNCS, vol. 6852, pp. 91–102. Springer, Heidelberg (2011)