cFireworks: a Tool for Measuring the Communication Costs in Collective I/O

Kwangho Cha
National Institute of Supercomputing and Networking,
Korea Institute of Science and Technology Information,
Daejeon, KOREA
Email: khocha@kisti.re.kr

Abstract—Nowadays, many HPC systems use the multi-core system as a computational node. Predicting the communication performance of multi-core cluster systems is complicated job, but finding out it is important to use multi-core system efficiently. In the previous study, we introduced the simple linear regression models for predicting the communication costs in collective I/O. In the models, however, because it is important to get the communication characteristics of the given system, we designed cFireworks, an MPI application to measure the communication costs of HPC systems. In this paper, we explain the detail concept and experimental results of cFireworks. The performance evaluation showed that the expected communication costs with the linear regression models generated by using the output of cFireworks are reasonable to use.

Keywords—Collective I/O; Communication Costs; Parallel Computing; Parallel I/O

I. INTRODUCTION

Because modern HPC systems consist of multi-core computational nodes, the systems frequently issue the complex intra-node and inter-node communications. In such systems, predicting the communication performance is difficult, but it is an important process to use HPC systems efficiently.

Collective I/O is the specialized I/O which provides the functions of single-file based parallel I/O. As the number of processes and the size of a problem increase, the importance of collective I/O is also emphasized. The most well known parallel programming library, the message passing interface (MPI), also supports collective I/O and it follows the two-phase I/O scheme in order to improve the collective I/O performance[1], [2], [3], [4]. The two-phase I/O consists of data exchange phase and I/O phase. In terms of data exchange phase, it has to generate a number of complicated communication operations and they become some parts of collective I/O overheads.

In the previous study[5], we have shown it is possible to improve the performance of collective I/O by reducing the communication costs. Furthermore, we also have demonstrated that finding out the expected communication costs before launching an application is important to reduce the communication costs in collective I/O. We used the linear regression models for predicting the communication costs and it was important to understand the communication characteristics of given systems in order to get the reasonable linear regression model. For this reason, we considered making cFireworks, an MPI application to measure the communication characteristics of multi-core cluster systems and partially introduced the basic concept of cFireworks in the previous work[5]. In this paper, we explain the more detail and improved concept of cFireworks and draw the experimental results with different kinds of multi-core cluster systems.

This paper is organized as follows. The previous research on communication model is summarized in Section II. Section III presents the main concept of cFireworks. The results of performance evaluations are described in Section IV. Finally, the conclusions are presented in Section V.

II. COMMUNICATION MODEL

When someone want to understand the process of communications or communication costs, it is helpful to use a valid communication model. In this section, we explain some communication models, such as the classical one and the linear regression model for collective I/O communications.

The LogP model is very well-known communication model which uses four parameters: L, o, g, and P stand for latency, overhead, bandwidth, and processors respectively[6][7]. It assumes a message passing procedure in distributed memory system and is intended for short messages. Many variants of LogP have been introduced as the system environments change[8][9].

Nowadays, many HPC systems use the multi-core system as a computational node. Communications in multi-core cluster systems are classified into two groups: intra-node and inter-node communications. In those multi-core cluster systems, because each core can communicate simultaneously, the communication media should be shared. Vienne et al.[10] suggested a predictive model for concurrent communication in multi-core systems. It sets several elementary sections of conflict parts and gets the communication time by predicting the cost of each section.

In some case, such as collective I/O, it is possible to expect the communication costs involving all processors by obtaining the communication time in the bottlenecked computational node[5]. Especially, data exchange time in collective I/O is proportional to the communication time in the hot-spot node. The simple linear model which uses the number of intra- and inter-node communications was introduced in order to expect the communication time in a node. The primary role of the prediction function in the study was predicting the relative performance of a given node set rather than obtaining accurate performance of the set. For this reason, they used a simple and
intuitive approach. The data exchange time in node $n_i$ can be described as:

$$T_{n_i}(ca_i, ce_i) = \alpha \cdot ca_i + \beta \cdot ce_i + \gamma$$  \hspace{1cm} (1)

where $ca_i$ is the number of intra-node communications within $n_i$ and $ce_i$ is the number of inter-node communications of $n_i$.

III. cFireworks

In the previous study, we discovered that the data exchange time of collective I/O was determined by the communication time of the most overloaded node. Furthermore the communication time is represented by $\alpha$, $\beta$ and $\gamma$ in equation (1). Because these values are related with the characteristics of the given system and communication procedures, it is necessary to identify the communication characteristics of the given system. For this reason we created a test program called cFireworks, in order to measure the appropriate communication parameters for the system.

Figure 1 shows the basic concept of the cFireworks test. In the first version of cFireworks, a process acts as a hot spot. In the real world, however, some processes in the same node can concurrently participate in the intra- and inter-node communications. For this reason, we designed the second version of cFireworks reflecting this situation. In the modified version, cFireworks has multiple hot spot processes. The processes are assigned to sub-groups and the processes send or receive data to their hot spot process in the sub-group. In this way, the program generates multiple concurrent communications in a node.

Algorithm 1 explains the pseudo code of cFireworks. It measures the communication time of a node by varying the number of intra- and inter-node communications (line 2, 3, 16, and 17) and the communication times with each number of communication pair are measured in every iteration.

There are two kinds of procedures to post asynchronous communications. In case of the first procedure intra-node communications are posted first (line 5 and 9), while the second procedure issues inter-node communications first instead of the intra-node ones (line 19 and 23). In other words, in the first measurement method, it generates the intra-node communications and then launches the inter-node communications; whereas in the second method, the inter-node communications are called first instead of the intra-node communications. In many cases, calling the intra-node communications first shows slightly better performance.

IV. PERFORMANCE EVALUATION

All experiments in this study were performed with Tachyon cluster systems\(^1\). Table I describes the specifications of Tachyon I and II system. A computational node of Tachyon I has four quad core CPUs, AMD’s Barcelona. Each CPU is equipped with 2 Mbytes L3 cache memory, DDR2 memory controllers and HyperTransport controller. Tachyon II is equipped with Intels Nehalem CPU which has an 8 Mbytes shared cache memory and DDR3 memory controllers.

A. Results of the cFireworks tests

Figures 2, 3, and 4 show the results of the cFireworks in the Tachyon I and II cluster system with a message size of 4 Mbytes. In order to reduce the number of iterations, cFireworks measures the communication time with a pair of intra- and inter-node communications. For this reason we created a test program called cFireworks, in order to measure the appropriate communication parameters for the system.

\(^{1}\)They are KISTI’s fourth supercomputers and the phase I system is ranked at 130 in the list of TOP500 most powerful supercomputers published in June 2008, and the phase II system is ranked at 14 in the list released in November 2009[11].
TABLE I: Specifications of KISTI Tachyon cluster systems

<table>
<thead>
<tr>
<th>Hardware</th>
<th>Tachyon I</th>
<th>Tachyon II</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>AMD Opteron 2.3GHz</td>
<td>Intel Xeon 2.93GHz</td>
</tr>
<tr>
<td>No. of nodes</td>
<td>188</td>
<td>3,176</td>
</tr>
<tr>
<td>No. of CPU cores</td>
<td>3008</td>
<td>25,408</td>
</tr>
<tr>
<td>No. of CPU cores/node</td>
<td>16</td>
<td>8</td>
</tr>
<tr>
<td>No. of CPU sockets/node</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>Socket to socket bandwidth</td>
<td>8GB/s</td>
<td>25.6GB/s</td>
</tr>
<tr>
<td>Memory</td>
<td>32GB/node</td>
<td>24GB/node</td>
</tr>
<tr>
<td>Interconnection network</td>
<td>InfiniBand 4× DDR</td>
<td>InfiniBand 8× QDR</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Software</th>
<th>Tachyon I</th>
<th>Tachyon II</th>
</tr>
</thead>
<tbody>
<tr>
<td>OS</td>
<td>CentOS 4.6</td>
<td>RedHat Enterprise 5.3</td>
</tr>
<tr>
<td>MPI</td>
<td>MVAPICH2 1.4</td>
<td></td>
</tr>
<tr>
<td>File System</td>
<td>Lustre 1.6.6</td>
<td>Lustre 1.8.1.1</td>
</tr>
<tr>
<td>Queue Scheduler</td>
<td>SGE 6.1u5</td>
<td>SGE 6.2u5</td>
</tr>
</tbody>
</table>

Algorithm 1 cFireworks algorithm

1: procedure INTRA_FIRST ▷ Intra-node communication first
2:    for x = 0; x < halfstar; x++ do
3:        for y = 0; y < halfstar; y++ do
4:            ▷ increase the no. of inter-node comm.
5:        end for
6:    end for
7: end procedure

8: procedure INTER_FIRST ▷ Inter-node communication first
9:    for x = 0; x < halfstar; x++ do
10:       for y = 0; y < halfstar; y++ do
11:           ▷ increase the no. of inter-node comm.
12:       end for
13:    end for
14: end procedure

Inter-node communications. That is, the hot spot process in Fig. 1 has the same number of ingress links and egress links for intra- or inter-node communications, respectively. For this reason, we have used a linear regression model obtained from the measured data considering equation (1) in order to cover every possible number of communications in a node. Figure 2a, 3a, and 4 illustrate the regression models derived from the data: the values of their coefficient of determination, $R^2$, are approximately 0.98s.

In case of Tachyon I, Figs. 2 and 3 show that the increasing rates of the communication time had altered when there were more than two pairs of intra-node communications. That is, when the number of intra-node communications is in the range of 2 and 7, the graph shows the rapid increases in communication time unlike the results between 0 and 2. We checked the system throughput with the measured data and could find that when the number of intra-node communications was less than 2, the throughput of the node still increased. If, however, it was more than two, the throughput remained steady and didn’t increase further. Consequently, the condition of that the number of intra-node communications reaches two is a criterion to determine whether the throughput of a node is saturated or not. For this reason, we’ve split the linear regression model into two variants: one for when throughput of the node is not saturated and another for when the throughput is saturated. By subdividing the regression model, the correctness of the model is improved. For example, when the number of intra-node communications is in the range of 2 and 7, $R^2$s are approximately 0.99s.

B. Validation test for cFireworks

In this section, we introduce the results of validation tests. The results of cFireworks were used for predicting the communication costs of collective I/O. In order to generate collective I/O workload, we used the MPI-Tile-I/O benchmark[12] and validated whether the linear regression models can provide a good indicator or not by comparing the execution time of MPI-Tile-I/O and the results of cFireworks. In the test, a 4×4 array was distributed to 16 processes, which wrote and read an 1 GB file. If the selected nodes have the different number of processes, the communication times in collective I/O are different according to the sequence of the nodes[5]. The performance was measured using four types of node sets that had 16 processes from the eight nodes as described in Table II and Figure 5.

Figure 6 shows the communication cost of the MPI-Tile-I/O and the expected values obtained by the linear regression models. In order to focus on the data exchange phase itself, the execution time without the file I/O phase was measured. In terms of collective I/O, if the size of I/O request is larger than the collective buffer size, collective I/O iterates the data exchange and I/O phases multiple times. We assumed that the data exchange time for a single iteration is proportional to the entire data exchange time and the linear regression models are used for predict the time for a single iteration. This is the reason why there is a gap between the measured data and the predicted ones in those figures.

3In most of MPI library, the write and read operations have the same communication workloads in the data exchange phase; however, unlike the read operation, the write operation has additional routines for post write and read modify write. Therefore, this causes the write operation to use more time than the read operation.
Fig. 2: Results of the cFireworks and their linear regression models (Tachyon I, intra-node communication first)

(a) \( T_{c}(p_a, p_r) = 0.007242p_a + 0.006617p_r + 0.005682 \)
\[ R^2 = 0.945452 \quad (0 \leq p_a < 2) \]
\[ R^2 = 0.979298 \quad (0 \leq p_a \leq 7) \]

(b) \( T_{c}(p_a, p_r) = 0.003805p_a + 0.007712p_r + 0.005866 \)
\[ R^2 = 0.990085 \quad (0 \leq p_a < 2) \]

(c) \( T_{c}(p_a, p_r) = 0.008372p_a + 0.006121p_r + 0.001588 \)
\[ R^2 = 0.996952 \quad (2 \leq p_a \leq 7) \]

Fig. 3: Results of the cFireworks and their linear regression models (Tachyon I, inter-node communication first)

(a) \( T_{c}(p_a, p_r) = 0.007335p_a + 0.006548p_r + 0.005723 \)
\[ R^2 = 0.942553 \quad (0 \leq p_a < 2) \]
\[ R^2 = 0.980699 \quad (0 \leq p_a \leq 7) \]

(b) \( T_{c}(p_a, p_r) = 0.003703p_a + 0.007509p_r + 0.006459 \)
\[ R^2 = 0.987416 \quad (0 \leq p_a < 2) \]

(c) \( T_{c}(p_a, p_r) = 0.008450p_a + 0.006095p_r + 0.005872 \)
\[ R^2 = 0.996952 \quad (2 \leq p_a \leq 7) \]
TABLE II: Test cases for the evaluation of the prediction functions

<table>
<thead>
<tr>
<th>Tests</th>
<th>Node set</th>
<th>Expected Communication Costs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Tachyon I</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Intra-node comm. first</td>
</tr>
<tr>
<td>T16-01</td>
<td>{4,4,2,2,1,1,1}</td>
<td>0.02138</td>
</tr>
<tr>
<td>T16-02</td>
<td>{1,1,1,1,2,4,2}</td>
<td>0.04051</td>
</tr>
<tr>
<td>T16-03</td>
<td>{1,1,2,2,1,1,1}</td>
<td>0.05213</td>
</tr>
<tr>
<td>T16-04</td>
<td>{1,1,1,1,4,2,2}</td>
<td>0.03471</td>
</tr>
</tbody>
</table>

Fig. 5: Data distribution of each test cases in Table II

As seen in Table II and Fig. 6, the prediction values and measured data of Tachyon II are much less than those of Tachyon I. That is, the communication costs of Tachyon II are lower than those of Tachyon I because the communication performance of Tachyon II is much higher.

The result of the experiment also demonstrates that the regression model can provide reasonable predictions in general. As seen in Table II, we used four kinds of test sets for the experiments. Because each node set has the different order of nodes communication patterns in collective I/O are also changed. In other words, each test case has the different number of intra- and inter-node communications in a hot spot node and this hot spot node determines the communication time of collective I/O. We input the number of communications in hot spot node of each test into our regression model and compared the results with the measured data.

The experimental results in Fig. 6 showed that our regression model could generate the reasonable prediction values. Because the predicted values are proportional to the real measured data in a greater or less degree, it is possible to use our regression model as a prediction model which can find a good node set without MPI execution. The performance differences among node sets in Tachyon II are not significant but the linear regression model still can tell the expected communication performance of Tachyon II.

V. CONCLUSION

Although predicting the communication performance of multi-core cluster systems is troublesome task, finding out the expected communication performance is important. In this study, we introduced cFireworks, an MPI application to measure the communication costs of HPC systems and the outputs of cFireworks were used for generating the linear regression models for predicting the communication costs. The results of performance evaluation showed that the expected communication costs with the linear regression models are reasonable to use. Furthermore, they also proved that cFireworks.
is simple and intuitive to use and helpful to generate linear regression models.

REFERENCES


