Designing an Efficient Simulation Tool for MPI Runtime Algorithm on Grid

Theewara Vorakosit and Puchong Uthayopas
High Performance Computing and Networking Center
Faculty of Engineering, Kasetsart University
Bangkok, Thailand
{g4685034,pu}@ku.ac.th

Abstract—For large computational grid system, MPI is one of the widely used programming models that can effectively harvest the power of massive number of computing system available. Nevertheless, smart algorithm is needed for some commonly used operation such as collective communication. This is due to many issues arise in large grid environment such as the unbalance network bandwidth, special network topology, and heterogeneity in term of hardware used. In this paper, we proposed an approach in modeling computational grid system using multi-level network. Then the problem of effective algorithm for multicasting in heterogeneous grid is studied. Simulator is used as a tool to evaluate the effectiveness of various scheme proposed. The study shows that the proposed model provides a powerful and easy way to evaluate and find a good algorithm that enable an efficient implementation of MPI runtime for Grid environment.

Keywords—Grid, simulation, MPI, multicast algorithm

I. INTRODUCTION

Grid is a distributed environment composed of large number of interconnected computing system. One of a widely used tool to harvest these abundant computing power is MPI over the grid. Performance of MPI runtime system relies heavily on the collective communication, while, in turn, hand on multicast algorithm used. Many algorithms for multicasting that used to be optimal under a single cluster environment such as binomial tree algorithm do not give an optimal solution. There is a need to discover several new heuristics that perform better in this new environment. The testing of newly proposed algorithms is usually hard or impossible due to two factors. First, Grid system is very dynamic. So, it is hard to repeat the experiments. Second, the usual size of the grid test-bed is too small to observe the problems such as scalability of the algorithm used. Thus, the simulator is a valuable tool that can be used to obtain the insight of grid behavior and evaluate the performance of various heuristic algorithms.

This paper presents the approach and tool that model the behavior of MPI runtime under grid environment. Our approach is composed of two models: a physical grid model and multicast operation model. Physical grid model is based on graph model while the multicast operation model is based on message scheduling and decision tree. An efficient simulator is designed according to the proposed model. The simulation library is designed reflect to the model. It is a small but efficient simulation library. The simulation tool then is used to develop a new multicast algorithm over Grid system.

The paper is organized as follows; Section II discusses about the previous works related to this paper. Section III gives the description of both physical grid model and multicast operation model. Section IV gives a design and architecture of TreeSim, the simulator library for the proposed model. Section V illustrates a performance of TreeSim library as well as various algorithms’ performance. Finally, conclusion and future works are described in section VI.

II. RELATED WORK

For the Grid system, finding the algorithm to build optimal broadcast tree is an NP-complete problem. Bernaschi and Iannello [1] proposed a broadcast algorithm based on α-tree. However, the value α must be specified before the α-tree is generated. Kielmann et al. [5] proposed the use of parameterized LogP model to address the problem. This model defines five parameters; number of processors (P), end-to-end latency (L), send overhead os(m), receive overhead or(m) and gap g(m). These parameters are now defined as a function of the message size m. The broadcast algorithm splits large message size M into k segment of size m. The goal is to find a tree shape and a segment size m that minimizes the completion time. Grid is then model as two-layer system. The coordinator nodes which are the representative of each cluster will participate in the wide-area broadcast first. Then, they forward the message inside their clusters to the target node. A different approach is proposed by Vorakosit and Uthayopas [11]. In that work, Grid is modeled as a graph and the efficient multicast tree is then derived using a genetic algorithm approach called GADT. Although is efficient, it required a substantial significant computation time. Thus, a new heuristic algorithm called LPBF[12] is proposed as a practical and fast algorithm for multicast over the wide area network.

These research works on algorithms require an efficient simulator for performance evaluation. There are many related simulator tools such as NS2 [8], Ginets [9], Ssfnet[10], GloMoSim[3] and cnet[4]. However, all this
tool focuses mostly on network level behavior. Although giving very detailed information, network simulator presented too low-level abstract to gain a true understanding of MPI operation. In this work, an appropriate level of abstraction is modeled such that the tool that is developed can give much more insightful information.

III. MODELLING THE GRID

A Grid system is represented as a weighted graph $G(V,E)$, where $V$ is a set of clusters and directed edge $E$ is a set of interconnection links between each cluster. Each edge has a weight $b_i$ which is the bandwidth available for the Grid application. This graph is represented by a bandwidth matrix. A bandwidth matrix ($B$) is defined as an $n \times n$ matrix, where $n$ is a number of clusters in the Grid. Each element $b_{ij}$ is a link bandwidth from cluster $i$ to cluster $j$. The diagonal values of the bandwidth matrix represent internal bandwidth of each cluster. Figure 1 and Figure 2 show a Grid graph and a bandwidth matrix, with respectively.

![Figure 1. Grid graph](image1)

$$
\begin{bmatrix}
1000 & 285 & 35 & 527 & 437 \\
285 & 1000 & 209 & 763 & 691 \\
35 & 209 & 1000 & 132 & 191 \\
527 & 763 & 132 & 1000 & 806 \\
437 & 691 & 191 & 806 & 1000 \\
\end{bmatrix}
$$

![Figure 2. Bandwidth matrix](image2)

When Grid environment is used for high performance computing platform, an application consists of many processes spread on nodes in Grid cooperate together. In this paper, we define process as a logical element that can execute a task. The definition of task is defined later in this section. The process can execute only one task at a time. The process must complete its task, either sending or receiving, and then it can execute another task. Processes are grouped into a set called Process Set (PS). We assume that the mapping between each process in PS to node is known prior to multicast operation. Process mapping function is a one-to-one correspondence function from process number to node in cluster. From the definition, there is only one process mapped to each node.

Multicast operation in Grid environment consists of more than one processes interact with each others. For a given multicast operation, there are one or more instances to complete the operation. Each instance’s solution of multicast operation solution is called multicast schedule. Multicast schedule is composed of two parts: multicast topology and multicast order. Multicast topology is the organizations of the sending/receive receiving operations among processes. It can be represented using tree, called multicast tree. Since the bandwidth of each link does not equal, the different order of transmission can result in a substantially different overall transmission time. This leads to the second part of the multicast schedule, the multicast order, which defines the dependency or order of the transmission. In each multicast schedule, a node can execute the $i^{th}$ step operation if and only if it has already finished the $(i-1)^{th}$ step. In each step, a sender selects a receiver based on its priority. Figure 3 shows an example of multicast schedule. The link label is a priority based on its source. For example, node 0 have three outgoing links, $(0,7)$, $(0,1)$, and $(0,4)$ that have priority of 2, 1, and 3, with respectively.

![Figure 3. Multicast schedule](image3)

In programmatic view, each link in Figure 3 can be considered as a task. A multicast task, or task, is defined as send/receiver operation where sender and receiver are nodes. Multicast schedule is a set of tasks. Task can be executed if and only if the sender already has data. At the beginning, only root process only has data.

IV. TReeSim ARCHITECTURE

To test the algorithm, the multicast tree simulator named TreeSim has been developed. Figure 4 shows the TreeSim architecture.

![Figure 4. The TreeSim architecture](image4)
From Figure 4, the simulator tool consists of two main modules: an algorithm front-end and an evaluator. The algorithm front-end generates multicast schedule from bandwidth matrix and grid information. Currently, TreeSim supports six algorithms: binomial, ECEF [2], ECEF2, MNF or Max-node-first, LPBF and GADT. Algorithm front-end produces and sends multicast schedule to an evaluator. The front-end receives total transmission time from the evaluator. After that, it may exit immediately or produce next input to evaluator depends on specific algorithm it is.

Evaluator function is to calculate the total transmission time for one instance of the multicast schedule. It uses bandwidth matrix, process mapping table, and multicast schedule as an input. The overall evaluator’s algorithm is shown in Figure 5.

The evaluator has three lists: start, running and finish. Each list keeps tasks in started, running and finished state, respectively. The runtime information of processes is maintained in a process list. The process list is derived from process set. It contains: id, free time, busy flag and data flag. A free time is a time that this node can start executing execute next task. A data flag is set to true if process already has data; otherwise it is set to false.

The overall algorithm is as followed: It gets a task from the start list. For each selected tasks, it checks whether source process already acquired data and whether source and destination processes are busy. If there is any condition does not satisfy these constrains, it skips the selected task, move the task back to the start list and continues to process the next task. If source and destination processes are in the same cluster, this kind of task is ready to finish. The transmission time is computed from bandwidth in that cluster.

The data flag for that destination process is set to true. Finish time of this task is calculated from transmission time plus sender’s free time. Free times of both processes are updated to finish time. Finally, this finished task is moved to finish list.

If source process and destination process are not in the same cluster, the algorithm is more complex. Based on its start time, it searches whether there is any finish task before the starting time. If so, the earliest finished task is moved to the finish list. This function is shown as FinishRunningTask in Figure 5. After that, this task is moved back to start list and the evaluator continues to get the next task from start list. This because process associated with this finish task must start its next task immediately. If there is no finish task, it updates bandwidth and estimated transmission time of tasks that use the same link as this current task. This is done by UpdateBandwidthAndTime. Then, this task is moved to running list.

When algorithm cannot get any tasks from the start list, it gets the earliest finished task from running list by calling FinishRunningTask function to move the finished task to finish list. After that it back to select tasks from the start list again. The algorithm ends when all tasks are moved to the finish list.

V. EXPERIMENTS AND DISCUSSION

TreeSim is used to conduct a comparative study for various multicast strategies over Grid. TreeSim support several algorithms including: Binomial Tree, ECEF, ECEF2, MNF, and LPBF. For the test system and data, a Grid of 8 to 64 clusters is used. The bandwidth matrix and number of nodes in each cluster were randomly generated. Simulated applications of size 8, 16, 32, ..., 2048 nodes are generated; for each problem instance, ten test cases are generated by randomly choosing the required number of nodes from Grid. The multicast message size used in all simulations is 1 Mb. We measured the time used in various steps in the simulator. The simulation was performed on a cluster of dual AMD AthlonMP 1800+ with 1024 MB SDRAM. First, algorithm decision times are measured.

Figure 6 compares the total transmission time of these five algorithms. Two-level algorithm yields good performance while one-level algorithm yields poor performance. This is a result of number of inter-cluster communication. Two-level algorithm optimizes inter-cluster communication which is the best utilization over slow link. So, they yield good performance.
This is a recursive algorithm, so it takes longer decision time. ECEF2, MNF and LPBF use heuristic algorithm to create inter-cluster communication from the delegated set of each cluster. These algorithms use binomial tree for intra-cluster communication. The worst decision time is ECEF algorithm because it is a one-level algorithm. Inter-cluster communication tasks are created from set of all nodes. The algorithm decision time depends mainly on the number of nodes. When number of nodes is fixed, the algorithm decision time vary slightly, as shown in Figure 8.

Figure 7 and Figure 8 show algorithm decision times that are used to create multicast schedule for the given system environment. The result shows that algorithm decision times increase when the number of nodes increases. The traditional binomial algorithm yields the best algorithm decision time because it depends on only node number. ECEF2 and MNF yield the same result. They are two-level algorithm. MNF yields more slightly algorithm decision time because it takes output from ECEF2 and reschedule based on number of nodes. LPBF yields more algorithm decision time than the first three algorithms. LPBF takes output from ECEF2 and reschedules based on branch length.

Figure 9 shows the evaluation time of five algorithms on 8 to 2048 nodes. The result shows that for one-level multicast algorithm, which are binomial and ECEF, the simulation time grows linearly when the number of node increases. While evaluation time of two-level algorithms, which are ECEF2, MNF and LPBF, yields a good simulation time. The time does not grow linearly as a function of nodes.

Figure 10 shows the result of evaluation time of five algorithms on 8 to 64 clusters. The result shows that when number of nodes is fixed to 1024 nodes, the simulation time depends on the number of clusters. The simulation time grows linearly as number of cluster increase. This is a result of inter-cluster communication time. In TreeSim, every time the task that involves inter-cluster communication is
moved to/from running list, TreeSim recomputes the total computation time. When number of clusters or number of nodes increase, the inter-cluster communication time increases that yield to an increase of evaluation time.

Figure 11 shows total running time of TreeSim. A total running time is an algorithm decision time plus evaluation time. As expected, ECEF2 and MNF have the shortest running time. LPBF is the third because its algorithm decision time is longer than ECEF2 and MNF. Then one-level algorithms have the longest running time due to very long evaluation time. From Figure 7 to Figure 10, we can conclude that the total running time depends mainly on inter-cluster communication which is the result from recomputation of total transmission time.

Figure 11. Total TreeSim running time for 16 clusters.

Figure 12. Total TreeSim running time for 1024 nodes

Figure 12 shows that when number of nodes is fixed, total running time increases when the number of clusters increases. Two-level algorithms still have better running time than one-level. ECEF2 and MNF have a same running time because MNF is based on ECEF2. LPBF is slower than ECEF2 and MNF when the number of nodes is less than 32. When the number of nodes is more than 32, LPBF has better running time. This because LPBF has longer algorithm decision time but shorter evaluation time on all number of clusters. Finally, one-level algorithms have poor running time due to their long evaluation time.

In general, from Figure 7 to Figure 12, we can conclude that performance of TreeSim depends mainly on inter-cluster communication. TreeSim can simulate up to few thousand nodes within about 15 minutes.

VI. CONCLUSION AND FUTURE WORKS

This paper presents an approach that can be used to efficiently model and simulate MPI multicast algorithm on Grid. A tool, called TreeSim, is developed based on the proposed model. This tool can simulate a very large Grid of a few thousand of nodes efficiently.

In the future, TreeSim will be adapted to handle more accurate model, for example, send/receive latency, multi-hop message transmission. Multi-hop message transmission is an important target to handle hierarchical network topology. Moreover, TreeSim can be integrated with MPI runtime system for runtime algorithm selection.

REFERENCES


