Understanding Torus Network Performance through Simulations

Sandeep Palur Dr. Ioan Raicu
Department of Computer Science, Illinois Institute of Technology, Chicago IL, USA
psandeep@hawk.iit.edu iraicu@cs.iit.edu

***Abstract* -- Technology developments in the storage and processing of data have spurred the development of distributed computing with distributed compute-clusters and supercomputers processing massive data that typically accompanies scientific experiments in the sciences. This has led to increasing demands for data transfers, with a requirement for high speed as well as requirements for Quality of Service, reliability, and security. These issues become more important in high-speed networks. A major consideration in the design of any parallel systems is the set of pathways over which the nodes communicate with each other. A Torus interconnect is a network topology for connecting processing nodes in a parallel computer system. A number of supercomputers on the TOP500 list use 3D Torus networks. In this work we benchmark the Torus network through appropriate performance metrics under different workloads using the ROSS(Rensselaer's Optimistic Simulation System)simulator. ROSS is a parallel discrete-event simulator that executes on shared-memory multiprocessor systems which is geared for running large-scale simulation models (i.e., supporting millions of object models is feasible). Through synthetic benchmarks, we have studied the communication imbalance generated by the common static single path routing in Torus interconnects. The long term goals are to demonstrate that multi-path dynamic routing could have significantly positive impact on both the end-to-end application performance as well as the aggregate system wide performance.**

I. Introduction

Supercomputers process demanding computational loads (process and data). It consists of numerous high performance processors for parallel processing. The processing power is paramount but the key aspect of parallel computers is the communication network that interconnects the computing nodes.

A. Torus Topology

Switch-less interconnection topology for connecting processing nodes in a parallel computer system. It can be visualized as a mesh interconnect with nodes arranged in a rectilinear array of N = 2, 3, or more dimensions, with processors connected to their nearest neighbors, and corresponding processors on opposite edges of the array connected. A Torus interconnect has a rich topology with many paths between any pair of nodes in the system. This configuration allows the addition of nodes to a system without degrading performance. Each new node is joined as an addition of a grid, linked to it with no extensive cabling or switches. It scales linearly, with little or no performance loss is strictly true for those problems that heavily rely on next neighbor communication. The addition of a node in a large system happens with much less working and potential troubles. Being the connections between nodes short and direct, the latency of the links is very low.

B. ROSS Simulator

ROSS is an acronym for Rensselaer's Optimistic Simulation System. It is a parallel discrete-event simulator that executes on shared-memory multiprocessor systems. ROSS is geared for running large-scale simulation models (i.e., 100K to even 1 million object models).The synchronization mechanism is based on Time Warp [2, 3, 4]. It uses a detection-and-recovery protocol to synchronize the computation. Any time an LP determines that it has processed events out of timestamp order, it "rollsback" those events, and re-executes them. ROSS was modeled after a Time Warp simulator called GTW or Georgia Tech Time Warp[5].

C. *CODES*

CODES is accurate and highly parallel simulation toolkit for exascale storage and is built on ROSS. CODES is divided into codes-base and codes-net. Codes-base is the utility library for construction of storage models and Codes-net is collection of network interconnect models and shared abstraction layer. CODES currently provide APIs for Torus and Dragonfly topology.

II.Experiments & Results

We ran experiments on 48 cores 250 GB ram machine with x86_64 architecture. We used ROSS simulator in parallel optimistic mode. Each server in the torus network communicates with its own pair. Server pairs are generated by Fisher–Yates shuffle algorithm. Each server sends and receives 100 messages.

In all experiments, we used the following configuration. Only the dimension length was varied from 2*2*2 to 16*16*16 and the values were extrapolated till 1 million based on the trend.

```
Packet Size="512 Bytes"
Modelnet="torus"
Message Size="2048 Bytes"
Dimension="3"
Dimension Length="X,X,X"
Link Bandwidth="2.0 GB"
Buffer Size="16384 Bytes"
Number of Virtual Channels="1"
Chunk Size="32"
```

The three major experiments we ran are as follows. We measured network metrics by

1. *Varying the size of the network:* As you can see from the below graphs the average throughput increases and number of hops increases with increase in the size of the network. It is evident from

the graphs that there are a lot of hot spots in the network

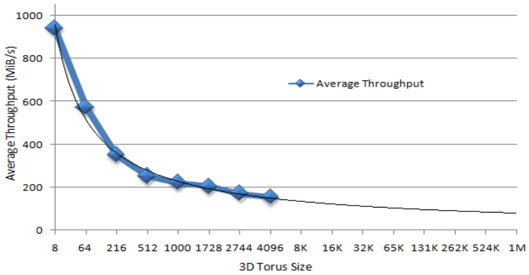

Figure 1: Average Throughput vs Network Size

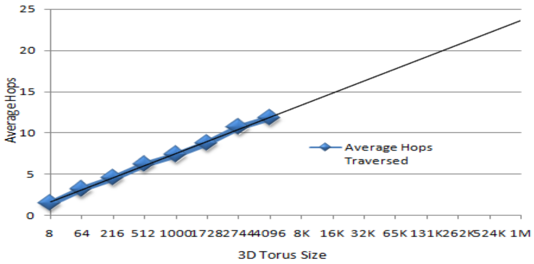

Figure 2 : Average Hops vs Network Size

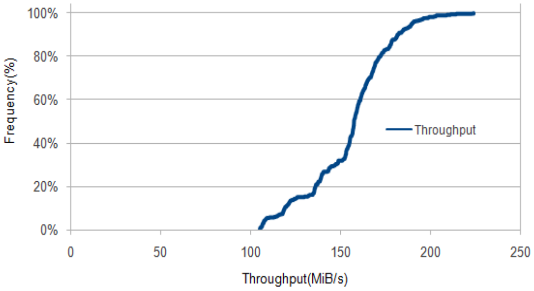

Figure 3 : CDF of Throughput on 4096 nodes(16*16*16)

2. *Varying the number of servers sending and receiving messages:* Graph clearly shows that average throughput decreases and difference between average and maximum latency increases with increase in the increase in number of servers transferring messages.

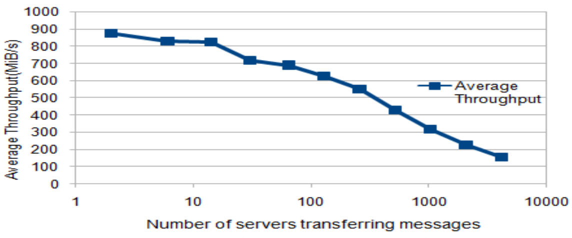

Figure 4 : Average Throughput vs Number of Servers Transferring Messages

3. *Varying the message size:* The average throughput and latency increases with increase in message size.

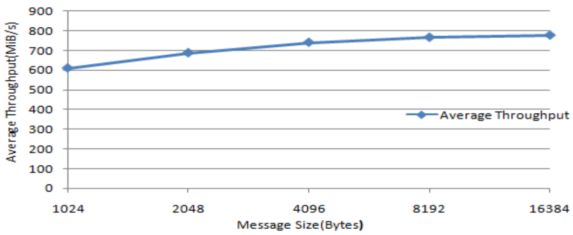

Figure 8 : Average Throughput vs Message Size

III . Conclusion and Future Work

Through synthetic benchmarks, we have studied the communication imbalance generated by the common static single path routing in torus interconnects. In torus network latency increases and throughput decreases as the size of the torus network and number of servers participating in message transfer increase. It is also evident from our experiments that throughput increases with increase in the message size. Since torus uses static single path routing, transferring messages between random server pairs leads to a lot of congestion at some intermediate nodes via which most of the messages pass through. These nodes become hot spots, reduce the throughput and increase latency. This leads us to believe that multi-path routing could have a positive impact on the performance of the network compared to the traditional static single-path routing.

The long term goals are as follows:

1. Design and develop a monitoring framework to monitor the network state and indicate the hot spots.

2. Demonstrate that multi-path dynamic routing could have significantly positive impact on both the end-to-end application performance as well as the aggregate system wide performance.

IV. References

[1] Narasimha R Adiga, Matthias A Blumrich, Dong Chen, Paul Coteus, Alan Gara, Mark E Giampapa, Philip Heidelberger, Sarabjeet Singh, Burkhard D Steinmacher-Burow, Todd Takken, et alBlue Gene/L torus interconnection network. IBM Journal of Research and Development.

[2] D. R. Jefferson and H. Sowizral. Fast concurrent simulation using the Time Warp mechanism,part I: Local control. Technical Report N-1906-AF, RAND Corporation, December 1982.

[3] D. R. Jefferson. Virtual time. *ACM Transactions on Programming Languages and Systems*, 7(3):404–425, July 1985.

[4] R. M. Fujimoto. Parallel discrete-event simulation. *Communications of the ACM*, 33(10):30–53,October 1990.

[5] S. Das, R. Fujimoto, K. Panesar, D. Allison, and M. Hybinette. GTW: A Time Warp system for shared memory multiprocessors. In *1994 Winter Simulation Conference Proceedings*, pages 1332–1339, December 1994.