A User-level Multicast Performance Comparison of Scalable Coherent Interface and Myrinet Interconnects

Sarp Oral and Alan D. George
{oral, george}@hcs.ufl.edu

High-Performance Computing and Simulation (HCS) Research Lab, Department of Electrical and Computer Engineering, University of Florida, P.O. Box 116200, Gainesville, Florida 32611, USA

Abstract

This paper compares and evaluates the multicast performance of two of the most widely deployed System-Area Networks (SANs), Dolphin's Scalable Coherent Interface (SCI) and Myricom's Myrinet. Both networks deliver low latency and high bandwidth to applications, but do not support multicast in hardware. We compared SCI and Myrinet in terms of their user-level performance using various software-based multicast algorithms under various networking and multicasting scenarios. The strengths and weaknesses of each network are comparatively presented in terms of numerous metrics, such as multicast completion latency, CPU utilization, link concentration and concurrency.

Keywords: Scalable Coherent Interface, Myrinet, multicast communication, performance evaluation, benchmarking.

1. Introduction

Due to advancements in VLSI technology, it has become possible to fit more transistors, gates, and circuits in the same die area, operating at much higher speeds. These enhancements have led the performance of microprocessors to increase steadily. Following this trend, commercial off-the-shelf (COTS) PCs or workstations built around mass-produced, inexpensive CPUs have become the basic building blocks for parallel computers instead of expensive and special-built supercomputers. Mass-produced, fast, commodity network cards and switches have allowed tightly integrated clusters of these workstations to fill the gap between desktop systems and supercomputers.

The use of these COTS technologies for both computing and interconnection systems has enabled scalable, high-performance parallel computing systems to be built at relatively lower cost than their custom-built counterparts. On the far end of this high-performance, low-cost interconnect spectrum lays System-Area Networks (SANs). A SAN is a low-latency, high-throughput interconnection network that uses reliable links to connect clustered computing nodes over short physical distances for high-performance parallel computing. Two of the recent and widely deployed SANs form very distinctive examples:

- Dolphin's Scalable Coherent Interface (SCI) [1, 2], which provides hardware-based distributed-shared memory over ring and torus-based topologies.
- Myricom's Myrinet [3], which provides message passing in a switched environment using a Network Interface Card (NIC) co-processor for off-loading work from the host processor.

The success of SAN-based, high-performance parallel computing depends on both the efficiency of the physical-layer components (switches, links, NICs, and the host architecture), and on the performance of the communication among the processors. The inter-process communication is often handled by collective communication operations in parallel applications [4]. Collective communication primitives play a major role in parallel programming by making the applications more portable among different platforms. Utilizing collective communication not only simplifies but increases the functionality and efficiency of the parallel tasks. As a result, efficient support of collective communication is important in the design of high-performance parallel systems.

Multicast communication is the basis of most collective communication operations. Its efficiency has a great impact on overall parallel system performance and is directly affected by its interaction with both the underlying network and topology. Therefore, investigating this relationship is essential for achieving high performance in parallel applications.

SCI and Myrinet are widely used for high-performance computing and both support reliable unicast communication in hardware, but they do not offer any hardware-based mechanism for multicasting. Their unicast performance has been investigated extensively both individually and in comparison to each other. This study will compare Dolphin's SCI and Myricom's Myrinet in terms of their user-level multicast performance. We have chosen various multicast algorithms from the literature and optimize them for each network to exploit their best features. These features include SCI's flexible point-to-point nature that readily supports torus topologies, and Myrinet's NIC processor for off-loading the host CPU.

The next section of this paper briefly describes the architecture of Dolphin's SCI and Myricom's Myrinet interconnects. In Section 3, a summary of related research

on SCI and Myrinet in the literature is provided, followed by Section 4 where we define the selected multicast algorithms to be used with the two networks. Section 5 defines three different communication schemes for various degrees of load sharing between the host and NIC co-processor on Myrinet. In Section 6 the comparative case study results are presented and analyses are rendered, followed by Section 7 where conclusions and future work are discussed.

2. Overview of the interconnects

2.1. Scalable Coherent Interface

SCI is an ANSI/IEEE standard [5]. SCI was initially aimed to be a very high-performance computer bus that would support a significant degree of multiprocessing. However, due to the technical limitations of "bus-oriented" architectures, the resulting ANSI/IEEE specification turned out to be a set of protocols that provide processors with a shared-memory view of buses using direct point-to-point links. Dolphin's SCI is a high-performance interconnect technology based on the IEEE SCI standard that addresses both the high-performance computing and networking domains [1, 2]. Emphasizing flexibility and scalability as well as multi-gigabit-per-second data transfers, Dolphin's SCI has found a main application area as a SAN for high-performance computing clusters.

The recent Dolphin SCI networks are capable of achieving low latencies (smaller than 2μs) and high throughputs (5.3 Gbps peak link throughput) over point-to-point links with cut-through switching. Figure 1 presents the architectural block diagram of Dolphin's PCI-based SCI NIC. Using the unidirectional ringlets as a basic block, it is possible to obtain a large variety of topologies, such as counter-rotating rings and unidirectional and bi-directional tori.

Unlike many other competing SANs, SCI also offers support for both the shared-memory and message-passing paradigms. By exporting and importing memory chunks, SCI provides a shared-memory programming architecture. All exported memory chunks have a unique identifier which is the collection of the exporting node's SCI node ID, and the exporting application's Chunk ID and the Module ID. Imported memory chunks are mapped into the importer application's virtual memory space. To exchange messages between the nodes, the data has to be copied to this imported memory segment. The SCI NIC detects this transaction, and automatically converts the request to an SCI network transaction. The PCI-to-SCI memory address mapping is handled by the SCI protocol engine. The 32-bit PCI addresses are converted into 64-bit SCI addresses, in which the most significant 16 bits are used to select between up to 64K distinct SCI devices.

Each SCI transaction typically consists of two sub-transactions, namely a request and a response. For the request sub-transaction, a read or write request packet is sent by the requesting node to the destination node. The destination node sends an echo packet to the requesting node upon receiving the request packet. Concurrently, the recipient node processes the request, and sends its own response packet to the requesting node. The requesting node will acknowledge and commit the transaction by sending an echo packet back to the recipient node up on receiving the response packet.

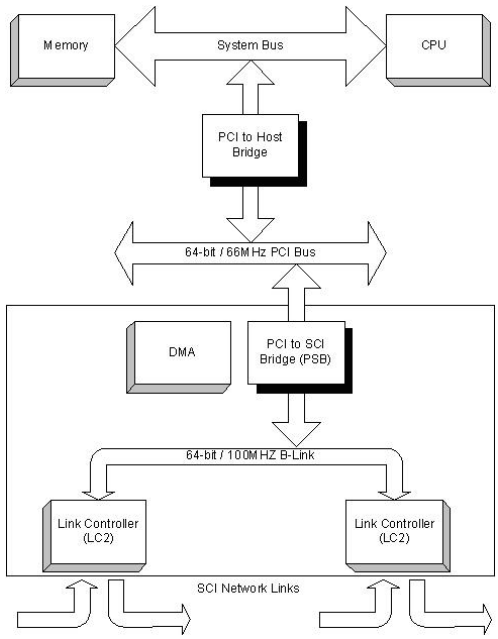


Figure 1: Architectural block diagram of SCI NIC (2D)

2.2. Myrinet

Myrinet [3] is the most widely deployed and arguably the most commercially successful high-speed interconnect for commodity clusters. Unlike SCI, Myrinet is a switch-based interconnect. Myrinet switches are based on a pipelined crossbar design and provide non-blocking, source-based, cut-through routing of packets with link-level flow control. Myrinet supports variable-length packets. The Myrinet NIC is connected to the PCI bus and has three DMA engines, a custom programmable RISC processor, and some amount of fast SRAM memory for data staging. Figure 2 illustrates the architectural block diagram of a PCI-based Myrinet NIC.

A Myrinet host adapter is based on the LANai chip, which integrates the RISC processor and the DMA engines. The Myrinet LANai processor operates at 133MHz and executes the Myrinet Control Program (MCP) stored on the on-board SRAM memory of the NIC. Recent Myrinet host adapters support 64-bit/133MHz PCI-X as well as 64-bit/66MHz PCI buses. Myrinet provides low latencies (as low as about 6μs) and high data rates (up to 2.0 Gbps).

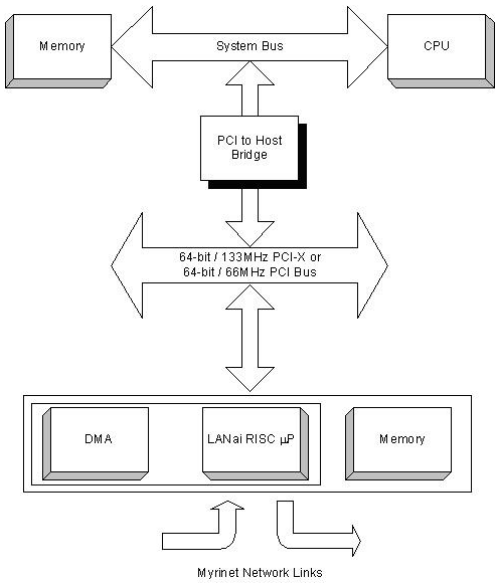


Figure 2: Architectural block diagram of Myrinet NIC

3. Related research

Multicast communication research in the literature can be briefly categorized into two groups, unicast-based and path-based [6]. Among the unicast-based multicasting methods, separate addressing is the simplest one [7]. Another approach for unicast-based multicasting is to use a multi-phase communication configuration for delivering the message to the destination nodes. In this method, the destination nodes are organized in some sort of binomial tree, and at each communication step the number of nodes covered increases by a factor of *n*, where *n* denotes the fan-out factor of the binomial tree. The U-torus multicast algorithm proposed by Robinson *et al.* [7] is a slightly modified version of this binomial-tree approach for direct torus networks that use wormhole routing.

Lin and Ni [8] were the first to introduce and investigate a path-based multicasting approach. Subsequently, path-based multicast communication has received attention and has been studied for direct networks [4, 7, 9]. Regarding path-based studies, this research will concentrate on the work of Robinson *et al.* [7, 9] in which the S-torus, M_d-torus, and M_u-torus algorithms are defined. These algorithms were proposed as a solution to the multicast communication problem for generic, wormhole-routed, direct unidirectional and bi-directional torus networks. More details about path-based multicast algorithms for wormhole-routed networks can be found in the survey of Li and McKinley [10].

SCI unicast performance analysis and modeling has been discussed in the literature [11-15], while collective communication on SCI has received little attention. Limited studies on this avenue have used collective communication primitives for assessing the scalability of various SCI topologies from an analytical point of view [16-17], except [18] has experimentally investigated the multicast performance of Dolphin/Scali interconnect for a 2D torus configuration.

NIC-based collective communication, in which the whole communication-related operations as well as the communication-related computational tasks are performed on the Myrinet NIC RISC processor, is widely studied in the literature for avoiding expensive host-NIC interactions and reducing system overhead and network transaction latency [19-23]. It was observed that, under such communication schemes, very low host CPU loads can be obtained at the cost of increased overall multicast completion latencies as the NIC processor is roughly a magnitude slower compared to modern host processors.

NIC-assisted multicasting was proposed as an answer to improve the multicast completion latencies of the NIC-based multicast communication schemes while obtaining similar degrees of host CPU loads for Myrinet. Bunitas *et al.* [24] presented a NIC-assisted binomial tree multicast scheme for Myrinet over Fast Messages from UIUC to improve the latency characteristics of NIC-based multicast algorithms.

Kurmann and Stricker [25] have comparatively evaluated the unicast performance of Dolphin's SCI and Myricom's Myrinet. They showed that both networks suffer performance degradation with non-contiguous data block transfers. Fischer *et al.* [26] also compared SCI and Myrinet. In their study, they concluded that, in terms of their performance and robustness analyses, Myrinet is a better choice compared to SCI.

This work compliments and extends previous work by providing comparative experimental evaluations of torus-optimized multicast algorithms for Dolphin's SCI versus various degrees of multicast work-sharing between the host and the NIC processors optimized for Myricom's Myrinet. Both for SCI and Myrinet networks, the multicast performance was evaluated using different metrics, such as multicast completion latency, root-node CPU utilization, link concentration and concurrency. The next two sections will provide details about the selected multicast algorithms for both networks, and the different multicast communication schemes evaluated for the Myrinet network.

4. Selected multicast algorithms

Different networks, topologies, and parallel applications create different communication scenarios. Some parallel applications are sensitive to message latency, while others are susceptible to variations in available bandwidth. Also, some multicast algorithms introduce more concurrency than others, resulting in reduced completion latencies but increased throughput requirements. Therefore, it is essential to have various multicast algorithms to fulfill these different priorities under various circumstances.

The simplest approach to multicast communication is *separate addressing*, in which a separate copy of the

message is sent directly from the source to every destination node. For small group sizes and short messages, separate addressing can be an efficient approach. However, for large messages and large group sizes, the iterative transmissions may result in large host-processor overhead. Another drawback of this protocol is linearly increasing multicast completion latencies with increasing group sizes.

An alternative to separate addressing is to use tree-based multicasting. The tree can be considered as a sequence of message-relaying steps. In the first step, the source node sends a single multicast message to only a subset of the destinations. In the next steps, each receiver node of the previous step relays the multicast message to its child nodes in its own subset. These message-relaying steps continue until every node in the destination set receives the multicast message successfully. Since several nodes communicate the message concurrently, this type of message-relaying communication greatly reduces the multicast completion latency compared to the separate addressing approach. There are different generic implementations of tree-based multicast algorithms, among which binomial tree, binary tree and serial forwarding are the most widely used.

Binomial tree provides the lowest latency and highest algorithmic concurrency for certain underlying topologies. In terms of latency, the binomial tree algorithm performs quite well on large groups with a suitable design because of its unbounded fan-out number and high concurrency. A *binary tree* is simply a special case of the binomial tree algorithm with a bounded binary fan-out. *Serial forwarding* is the simplest form of tree-based multicast communication where the multicast message is propagated from source node to all destination nodes in a chain-type communication. However, this simplicity results in no algorithmic concurrency and there can be only one message in transfer at any given time. The multicast completion latency increases linearly with the group size. These multicast algorithms, including separate addressing, are topology independent and applicable to all networks, so they form a suitable basis for comparing the performance over different SANs.

Throughout this work, the aggregate collection of all destination nodes and the source node is called the multicast group. Therefore, for a given group with size d, there are d-1 destination nodes.

U-torus [6] is a multicast algorithm that uses a binomial-tree approach to reduce the total number of required communication steps. For a given group of size d, the lower bound on the number of steps required to complete the multicast by U-torus will be $\lceil \log_2 d \rceil$. This reduction is achieved by increasing the number of covered destination nodes by a factor of 2 in each communication step. *S-torus* is a single-phase multicast routing algorithm, defined by Robinson *et al.* [9], in which the destination nodes are rank ordered to form a Hamiltonian cycle. The ranking of the node is based on their respective physical locations in the torus network. More detailed information about Hamiltonian node rankings can be found in [9]. The root node issues a single multicast worm which visits each destination node one after another following an ordered set. At each destination node, the header is truncated to remove the visited destination address and the worm is re-routed to the next destination. The algorithm continues until the last destination node receives the message.

Although simple, single-phase communication is known for large latency variations for large sets of destination nodes [27]. Robinson *et al.* proposed the multi-phase multicast routing algorithm, M-torus [9], to further improve the S-torus algorithm. The idea was to shorten the path lengths of the multicast worms to stabilize the latency variations and to achieve better performance by partitioning the multicast group. They introduced two variations of the M-torus algorithm, M_d-torus and M_u-torus. *M_d-torus* uses a dimensional partitioning method based on the respective sub-torus dimensions to eliminate the costly dimension-switching overhead. *M_u-torus* uses a uniform partitioning mechanism to equalize the partition lengths. In both of these algorithms, the root node separately transmits the message to each partition and the message is then further relayed inside the subsets using multicast worms. For a k-ary N-dimensional torus network, where k^N is the total number of nodes, the M_d-torus algorithm needs N steps to complete the multicast operation. M_u-torus is parameterized by the partitioning size, denoted by r. For a group size of d, the M_u-torus algorithm with a partitioning size of r requires $\lceil \log_r d \rceil$ steps to complete the multicast operation.

For our study we have used binomial and binary tree, separate addressing and serial forwarding multicast algorithms enhanced with various degrees of multicast work-sharing between the host and the NIC processors optimized for Myricom's Myrinet interconnect. For Dolphin's SCI we have used the separate addressing and the torus-specific U-torus, S-torus, M_d-torus, and M_u-torus multicast algorithms. The next section introduces the communication schemes that provide degrees of multicast work-sharing between the host and the NIC processors for Myricom's Myrinet interconnect.

5. Host processor vs. NIC processor multicasting

For interconnects with an onboard NIC processor, multicast communication primitives can be implemented at two different extremes: host-based and NIC-based. Between these two extremes lies another level of implementation: NIC-assisted multicasting.

Host-based multicast communication is the easiest and most conventional way of implementing a multicast communication primitive. In this scheme, the host processor handles all multicasting tasks, such as multicast tree creation, calculating the next set of destination nodes for each multicast step, and the issuing of the unicast send

and receive operations. This type of implementation introduces increased CPU load, resulting in a lower computation-communication overlap available for programs. However, as the fast host processor performs all the tasks, host-based multicasting provides small multicast-completion latencies.

In the *NIC-based* scheme, the NIC co-processor handles all multicasting tasks instead of the host processor. This scheme provides a reduced CPU load and high computation-communication overlap for parallel programs. The NIC processor is roughly an order of magnitude slower than modern host processors and performing all the tasks on this relatively slow processor increases multicast completion latency.

Between these two extremes, a compromise has been proposed to this problem, called *NIC-assisted* multicasting [23]. In this approach, work is shared between two processors, with the host processor handling computationally intensive multicast tasks, such as multicast tree creation, and the NIC processor handling communication-only multicast tasks, such as unicast send and receive operations. This solution is also found to present optimal multicast completion latency and provides moderate CPU loads, resulting in a modest computation-communication overlap available for programs.

6. Case study

The testbed on which the case study is performed consists of a 16-node Linux system. Each node features dual 1GHz Intel Pentium-III processors, 256MB of PC133 SDRAM, ServerSet III LE (rev 6) chipset, and a 133MHz system bus. For the Dolphin SCI experiments, Dolphin PCI-64/66/D330 SCI NICs with 5.3 Gb/s link speed are used along with Scali's SSP (Scali Software Platform) 3.0.1 communication software. The nodes are interconnected to form a 4×4 unidirectional 2D torus using 2m cables. The Myrinet experiments are performed using M2L-PCI64A-2 Myrinet NICs with a 1.28 Gb/s link speed along with Myricom's GM-1.6.4 communication software. The nodes are interconnected to form a 16-node star network using 16 3m Myrinet LAN cables and a 16-port Myrinet switch for the Myrinet experiments. Multicast-tree creation is removed from the critical path and performed at the beginning of each algorithm in every node to decrease the completion latencies. All of the multicast algorithms are evaluated in terms of their multicast completion latency, root-node CPU utilization, link concentration, and link concurrency, for group sizes of 4, 6, 8, 10, 12, 14, and 16 and for message sizes of 2B and 64KB. Each experiment is evaluated for each message and group size for 100 executions, where each execution has 50 repetitions. The variances of the multiple experiments are very small and averaged results are used. For each algorithm, the latencies are probed and measured separately. The maximum user-level host CPU utilization of the root node is measured using Linux's built-in `sar` utility. Finally, the link concentration and concurrency of each algorithm are calculated for each group size based on the communication pattern observed throughout the experiments.

Modified versions of S-torus, M_d-torus, and M_u-torus are used for the SCI experiments. These algorithms were originally designed to use multi-destination worms. However, as with most high-speed interconnects available on the market today, our testbed does not support multi-destination worms. Therefore, store-and-forward versions of these algorithms are developed. Partition length *r* of 4 is used for M_u-torus. The partition information for U-torus is embedded in the relayed multicast message at each step.

Binomial tree, binary tree, serial forwarding and separate addressing algorithms are developed for the host-based and the NIC-assisted communication schemes for the Myrinet experiments. All of our host-based multicast algorithms are designed in the user-level GM-1.6.4 provided by Myricom. For NIC-assisted multicast algorithms the provided Myrinet Control Program (MCP) is modified. MCP is firmware that executes on the LANai processor. The updated MCP added an extra 8μs to the unicast sends whereas the unmodified minimum one-way latency was measured as 17μs. Table 1 presents the pseudo-code of the NIC-assisted communication for the root node and the intermediate and the destination nodes. The italicized parts of the pseudo code are either performed or initiated by the host CPU and the rest is by the NIC processor. SDMA is the process of the host writing to NIC memory and signaling the NIC upon completion of the write operation. RDMA, on the other hand, is the process of the NIC writing to the host memory and signaling it upon the completion of the write operation.

Table 1: Pseudo-code for NIC-assisted communication

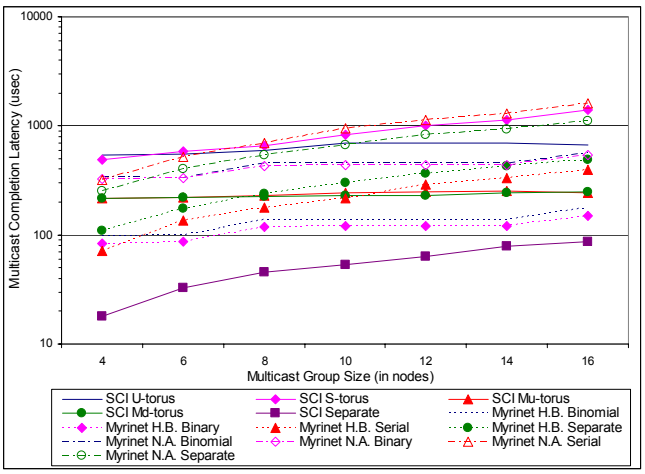
Root node
Obtain multicast host names *Obtain GM MCP base address pointer* *Build multicast tree* *SDMA* *Wait for completion* Do multicast RDMA
Intermediate and destination nodes
Listen for incoming multicast calls Receive message RDMA *Check multicast tree* *SDMA* Do multicast

The following subsections will present and analyze the experimental results obtained.

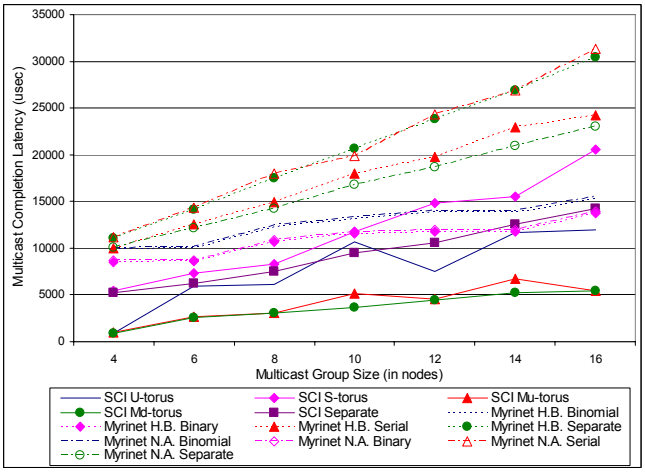
6.1. Multicast Completion Latency

Completion latency is an important metric for evaluating and comparing different multicast algorithms, as it reveals how suitable an algorithm is for a given network. Two

different sets of experiments for multicast completion latency are performed in this case study, one for a small message size of 2B, and another for a large message size of 64KB. Figure 3(a) illustrates the multicast completion latency versus group size for small messages, while Figure 3(b) presents the same for large messages for both networks combined with the various multicast algorithms. Figure 3(a) is presented with a logarithmic scale for clarity.



(a)



(b)

Figure 3: Small-message (a) and large-message (b) multicast completion latency versus group size, where H.B. and N.A. denote the host-based and NIC-assisted communication, respectively

Let us first bring our attention to the small-message multicast latencies in Figure 3(a). As explained previously, separate addressing is based on a simple iterative use of the unicast send operation. Therefore, for small messages the inherent unicast performance of the underlying network significantly dictates the overall performance of the multicast algorithm. This trait can be observed by comparing the small-message multicast completion latencies of SCI and Myrinet. SCI is inherently able to achieve almost an order of magnitude lower unicast latency compared to Myrinet. Simplicity and cost-effectiveness of the separate addressing algorithm for small messages, combined with SCI's unicast characteristics, result in the outcome where SCI separate addressing clearly performs the best compared to all other SCI and Myrinet multicast algorithms.

NIC-assisted Myrinet separate addressing does not provide a comparable performance level to the host-based version due to the costly SDMA and RDMA operations. It is observed that the SDMA and RDMA operations impose a significant overhead for small-message communications. Moreover, all three multicast schemes show a linear increase with increasing group size.

SCI S-torus is one of the worst performing algorithms next to the Myrinet NIC-assisted serial forwarding algorithm for small messages. Host-based Myrinet serial forwarding performs better compared to these two algorithms. The store-and-relay characteristic of serial forwarding algorithms result in no parallelism in message transfers and thus degrades the performance. Moreover, the expensive SDMA and RDMA operations cause the NIC-assisted serial forwarding algorithm to perform poorly compared the host-based version. As can be seen, all three multicast algorithms show a linear increase in multicast completion latency with respect to the increasing group size.

Unlike the separate addressing and serial forwarding algorithms of SCI and Myrinet, binomial and binary tree algorithms exhibit nearly constant completion latencies with increasing group sizes. Among these, Myrinet host-based binary and binomial tree algorithms perform best. Comparable algorithms, such as SCI U-torus, M_d-torus and M_u-torus algorithms show higher completion latencies. The difference is attributed to the low sender and receiver overheads of host-based Myrinet multicasting as well as the simplicity of the star network compared to the complex torus structure of the SCI network. For a given star network, the average message transmission paths are shorter compared to the same sized torus network. One other reason is increasing efficiency of the lightweight Myrinet GM message-passing library with increasing complexity of communication algorithms, compared to the shared-memory Scali API of the SCI interconnect. The effect of the expensive SDMA and RDMA operations can be clearly seen in the NIC-assisted Myrinet binary and binomial tree algorithms compared to their host-based counterparts.

For the large-message multicast latencies in Figure 3(b), the SCI algorithms appear to perform best compared to their Myrinet counterparts. This outcome is judged to be primarily due to the higher data rate of SCI compared to Myrinet (i.e. 5.3 Gb/s vs. 1.28 Gb/s). It should be noted that the Myrinet testbed available for these experiments is not quite representative of the latest generation of Myrinet equipment (which feature 2.0 Gb/s data rates). However,

we believe that our results would follow the same general trend for large messages on the newer hardware.

Among the SCI algorithms, M_d-torus is found to be the best performer. The M_d-torus and M_u-torus algorithms exhibit similar levels of performance. The difference between these two becomes more distinctive at certain data points, such as 10 and 14 nodes. For these group sizes, the partition length for M_u-torus does not provide perfectly balanced partitions, resulting in higher multicast completion latencies. For large messages, U-torus exhibits similar behavior to M_u-torus. S-torus is the worst performer compared to all other SCI multicast algorithms. Moreover, S-torus, similar to other single-phase, path-based algorithms, has unavoidably large latency variations due to the long multicast message paths [27].

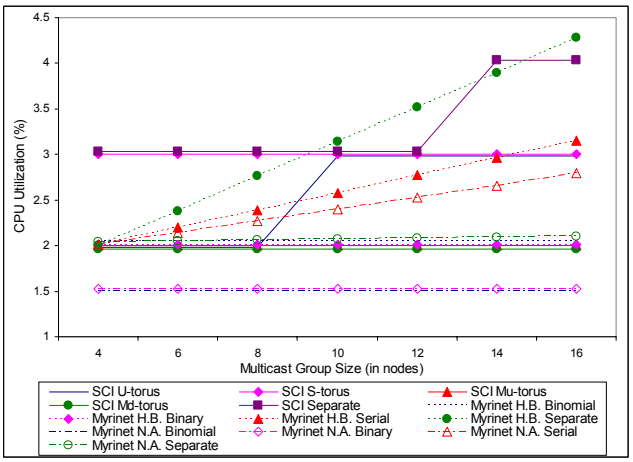
Myrinet multicast algorithms seem to be no match for the SCI-based ones for large messages. Unlike the small-message case, Myrinet NIC-assisted binary and binomial tree algorithms provide nearly identical completion latencies to their host-based counterparts for large messages. The SDMA and RDMA overheads are negligible for large messages and due to this reason NIC-assisted multicast communication performance is enhanced significantly. Moreover, NIC-assisted communication improves the performance of the separate addressing algorithm most, compared to all other Myrinet multicast algorithms, due to the relative reduction of the overall effect of the SDMA and RDMA overheads on the multicast completion latencies. By contrast, NIC-assisted communication degrades the performance of the Myrinet serial forwarding algorithm, because the SDMA and the RDMA overheads are incurred at each relaying node.

6.2. User-level CPU Utilization

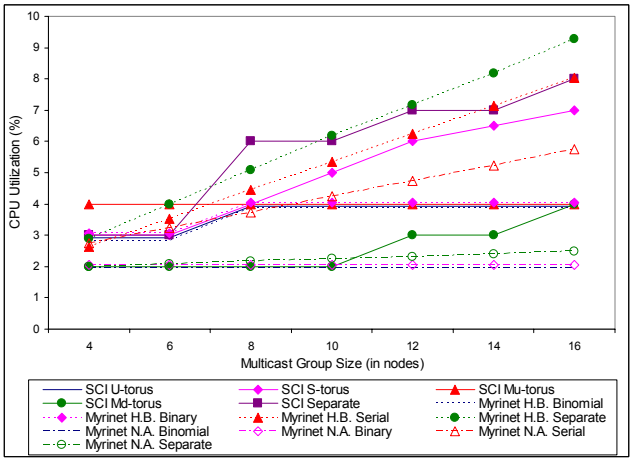
Host processor load is another useful metric to assess the quality of a multicast protocol. Figures 4(a) and (b) present the maximum CPU utilization for the root node of each algorithm. As before, results are obtained for various group sizes and for both small and large message sizes. Root-node CPU utilization for small messages are presented with a logarithmic axis for clarity.

For small messages, SCI M_d-torus and M_u-torus provide constant 2% CPU utilization for all group sizes. Both algorithms use a tree-based scheme for multicast, which increases the concurrency of the message transfers and decreases the root-node workload significantly. Also, it is observed that SCI S-torus exhibits relatively higher utilization compared to these two but at the same time provides a constant CPU load independent of group size. As can be seen, SCI U-torus exhibits a step-like increase for small messages due to the increase in the number of communication steps required to cover all destination nodes.

Myrinet host-based binomial and binary tree algorithms provide an identical CPU utilization to that of SCI M_d-torus and M_u-torus. Host-based separate addressing and serial forwarding algorithms both show a perfect linear increase in terms of small-message CPU utilization, where serial forwarding performs better compared to separate addressing.



(a)



(b)

Figure 4: Small-message (a) and large-message (b) user-level CPU utilization versus group size, where H.B. and N.A. denotes the host-based and NIC-assisted communication, respectively

Myrinet NIC-assisted binomial and binary tree algorithms lower the root-node CPU utilizations as expected. These two algorithms provide the lowest and a constant CPU utilization for all group sizes. As the number of root-node message transfers increases logarithmically, instead of linearly as in separate addressing, with increasing group sizes they are found to be independent of group size in these experiments. Similarly, NIC-assisted separate addressing lowers the CPU utilization compared to the host-based version, and provides a very slowly increasing utilization. Similar reduction is also observed for the NIC-assisted serial forwarding algorithm compared to its host-based counterpart. Unlike NIC-assisted separate addressing, serial forwarding cannot sustain this low utilization, and it

increases linearly with increasing group size. The linear increase of the serial forwarding is due to the ever extending path lengths with increasing group sizes.

Meanwhile, for large messages, it is observed that as group size increases the CPU load with the SCI S-torus algorithm also linearly increases. The SCI separate addressing algorithm has a nearly linear increase in CPU load for large messages with increasing group size, as well. By contrast, since the number of message transmissions for the root node is constant, M_d-torus provides a nearly constant CPU overhead for large messages and small group sizes. However, for group sizes greater than 10, the CPU utilization tends to increase due to variations in the path lengths causing extended polling durations. For large messages, SCI M_u-torus also provides higher but constant CPU utilization.

Host-based Myrinet binomial and binary tree algorithms provide similar levels of CPU utilization to SCI M_u-torus for large messages and large group sizes. Similar to the small-message case, host-based separate addressing and serial forwarding have linearly increasing utilizations with increasing group sizes, where serial forwarding performs better compared to separate addressing.

NIC-assisted binomial and binary tree algorithms again provide the smallest and sustainable constant CPU utilizations for large messages for all group sizes. Similar to the small-message case, separate addressing benefits most from NIC-assisted communication as can be seen from 4(b). Serial forwarding also exhibits lower CPU utilizations with the NIC-assisted communication.

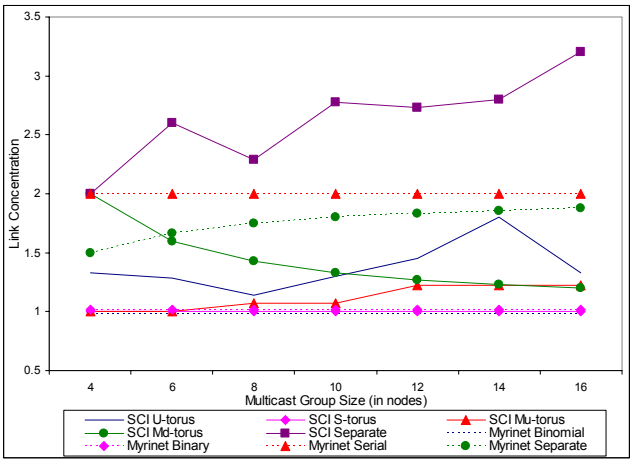
6.3. Link Concentration and Link Concurrency

Link concentration is defined here as the ratio of two components: the number of link visits and the number of used links. *Link visits* is defined as the cumulative number of links used during the entire communication process, while *used links* is defined as the number of individual links used. *Link concurrency* is the maximum number of messages that are in transit in the network at any given time. Link concentration and link concurrency are given in Figures 5(a) and 5(b), respectively. Myrinet host-based and NIC-assisted communication schemes have identical link concentration and concurrency, so they are not separately plotted. Link concentration combined with the link concurrency illustrates the degree of communication balance. The concentration and concurrency values presented in Figures 5(a) and 5(b) are obtained by analyzing the theoretical communication structures and the experimental timings of the algorithms.

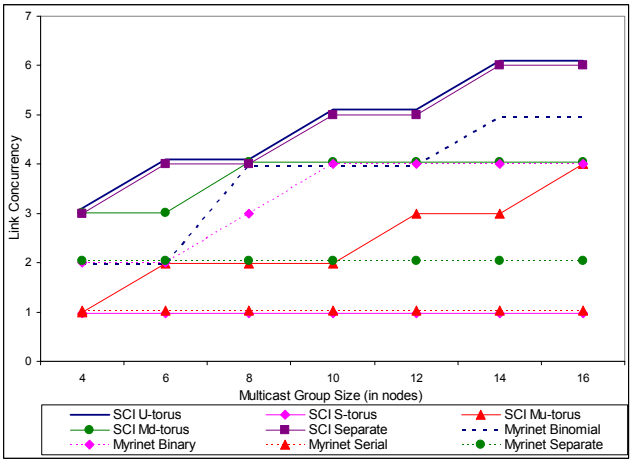
SCI S-torus is a simple chained communication and there is only one active message transfer in the network at any given time. Therefore, it has the lowest and a constant link concentration and concurrency compared to other algorithms. Due to the high parallelism provided by the recursive doubling approach, the SCI U-torus algorithm has the highest concurrency. SCI separate addressing exhibits an identical degree of concurrency to the U-torus, because the multiple message transfers overlap at the same time due to the network pipelining feature available over the SCI torus network. SCI M_d-torus has inversely proportional link concentration versus increasing group size. In M_d-torus, the root node first sends the message to the destination header nodes, and they relay it to their child nodes. As the number of dimensional header nodes is constant (k in a k-ary torus), with increasing group size each new child node added to the group will increase the number of available links. Moreover, due to the communication structure of the M_d-torus, the number of used links increases much more rapidly compared to the number of link visits with the increasing group size. This trend asymptotically limits the decreasing link concentration to 1. The concurrency of M_d-torus is upper bounded by k as each dimensional header relays the message over separate ringlets with k nodes in each. The SCI M_u-torus algorithm has low link concentration for all group sizes, as it multicasts the message to the partitioned destination nodes over a limited number of individual paths, where only a single link is used per path at a time. By contrast, for a given partition length of constant size, an increase in the group size results in an increase in the number of partitions as well as an increase in the number of individual paths. This trait results in more messages being transferred concurrently at any given time over the entire network.

The Myrinet serial forwarding algorithm is very similar to SCI S-torus in terms of its logical communications structure. Therefore, as expected it also exhibits a constant concentration. However, Myrinet serial forwarding has a higher link concentration compared to S-torus, and the difference is due to the physical structure of the two interconnects. In Myrinet, the degree of connectivity of each host is fixed at 1, whereas in SCI it is N for an N-dimensional torus system. Similar to S-torus, serial forwarding has the lowest link concurrency as well. Myrinet binomial and binary trees have identical link concentration to SCI S-torus. These two algorithms use a tree-based communication scheme and each node is visited only once, while there are no relaying nodes on the message path. The number of required links to establish a connection between any two nodes is 2 for a single-switch Myrinet network. Moreover, different than the other Myrinet algorithms, in separate addressing all communication is initiated from the same host, thus they all use the same link until they are routed at the switch to different destination nodes. Therefore, Myrinet separate addressing has an asymptotically upper bound link concentration at 2, with increasing group size. Myrinet serial forwarding has the lowest and constant link concurrency for all group sizes due to the reasons explained before. Myrinet separate addressing also has a constant but higher link concurrency. Myrinet binomial and binary tree algorithms have higher and variable link

concurrencies with respect to the group size. Binary tree has a bounded fan-out number which decreases the link concurrency compared to the binomial tree.



(a)



(b)

Figure 5: Link concentration (a) and link concurrency (b) versus group size

7. Conclusions

This paper evaluates and compares the user-level multicast performance of Dolphin SCI and Myricom Myrinet. These two are the most widely deployed interconnects for cluster computing, and they are known for their low-latency and high-bandwidth characteristics. Unfortunately, they do not support multicast in hardware, as it is the case with most of the SANs available today. SCI and Myrinet are compared in terms of their multicast performance using various software-based algorithms under various networking and multicasting scenarios. The multicast performance of each network is experimentally evaluated, and various strengths and weaknesses are identified and comparatively analyzed in terms of metrics including multicast completion latency, CPU utilization, link concentration and concurrency.

Multicast algorithms differ in their algorithmic and communication pattern complexity. Oftentimes, the functionality of the algorithms increase with complexity, but occasionally the increased complexity degrades the performance in some circumstances. For some cases, such as small-message multicasting for small groups, using simple algorithms helps to obtain the true performance of the underlying network. For example, due to its simplicity and the inherently lower unicast latency of SCI, the SCI separate addressing algorithm is found to be the best choice for small-message multicasting for small groups.

The lightweight GM software for message passing in Myrinet performs efficiently on complex algorithms. Therefore, while simple algorithms such as separate addressing perform better on SCI, it is observed that more complex algorithms such as binomial and binary tree achieve good performance on Myrinet for small-message multicast communication.

For large messages, Dolphin's SCI has a clear advantage due its higher link data rate compared to Myricom's Myrinet (i.e. 5.3 Gb/s of SCI vs. 1.28 Gb/s of Myrinet used in this study). Although the newest Myrinet hardware features higher data rates (i.e. 2.0 Gb/s) than our testbed, these rates are still significantly lower than SCI. Therefore, we expect that our results for large messages would follow the same general trend even for the newest generation of Myrinet equipment.

Myrinet NIC-assisted communication provides low host CPU utilizations for small and large message and group sizes. Not only complex algorithms such as binomial and binary tree, but also simple ones like separate addressing benefit significantly from this approach. However, multicast performance of NIC-assisted communication is directly affected by the cost of SDMA and RDMA operations. The overhead of these operations limits the potential advantage of this approach.

There are several possibilities for future research, one of which is integrating the selected algorithms with higher communication layers such as MPI and comparatively evaluating the high-level multicast performance of SCI and Myrinet interconnects. Yet another direction is to extend our SAN-based comparative research as a low-level communication service for other high-performance networks including use not only for cluster networks but also in hierarchical grid networks.

8. Acknowledgments

This research was supported in part by the U.S. Department of Defense and by matching funds from the University of Florida for the iVDGL project funded by the National Science Foundation. Further support was received in terms of SCI equipment from Dolphin Interconnect Solutions Inc., software from Scali Computer AS, and Myrinet equipment provided to us by Sandia National Labs.

9. References

[1] Dolphin Interconnect Solutions, PCI SCI Cluster Adapter Specification, 1996.

[2] D.B. Gustavson, and Q. Li, "The Scalable Coherent Interface (SCI)," *IEEE Communications*, Vol.34, No.8, pp. 52-63, August 1996.

[3] N. Boden, D. Cohen, R. Felderman, A. Kulawik, C. Seitz, J. Seizovic, and W. Su, "Myrinet: A Gigabit-Per-Second Local Area Network," *IEEE Micro*, Vol.15, No.1, pp. 29-36, February 1995.

[4] P.K. McKinley, Y.Tsai, and D.F. Robinson, "Collective Communication in Wormhole-Routed Massively Parallel Computers," *IEEE Computer*, Vol.28, No.2, pp. 39-50, 1995.

[5] IEEE, SCI: Scalable Coherent Interface, IEEE Approved Standard 1596-1992, 1992.

[6] Y. Tseng, D.K. Panda, and T Lai, "A Trip-Based Multicasting Model in Wormhole-Routed Networks with Virtual Channels," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 7, No.2, pp. 138-150, 1996.

[7] D.F. Robinson, P.K. McKinley, and B.H.C. Cheng, "Optimal Multicast Communication in Wormhole-Routed Torus Networks," *IEEE Transactions on Parallel and Distributed Systems*, Vol.6, No.10, pp. 1029-1042, 1995.

[8] X. Lin and L.M. Ni, Deadlock-Free Multicast Wormhole Routing in Multicomputer Networks, *Proc. of 18th Annual International Symp. on Computer Architecture*, Toronto, Canada, May 1991, pp. 116-124.

[9] D.F. Robinson, P.K. McKinley, and B.H.C. Cheng, "Path-Based Multicast Communication in Wormhole-Routed Unidirectional Torus Networks," *Journal of Parallel and Distributed Computing*, Vol.45, No.2, pp. 104-121, 1997.

[10] L.M. Ni and P.K. McKinley, "A Survey of Wormhole Routing Techniques In Direct Networks," *IEEE Computer*, Vol.26, No.2, pp. 62-76, 1993.

[11] K. Omang and B. Parady, "Performance Of Low-Cost Ultrasparc Multiprocessors Connected By SCI," Technical Report, Department of Informatics, University of Oslo, Norway, 1996.

[12] M. Ibel, K.E. Schauser, C.J. Scheiman and M. Weis, "High-Performance Cluster Computing using SCI," *Proc. of Hot Interconnects Symposium V*, Palo Alto, CA, Aug. 1997.

[13] M. Sarwar and A. George, "Simulative Performance Analysis of Distributed Switching Fabrics for SCI-Based Systems," *Microprocessors and Microsystems*, Vol.24, No.1, pp. 1-11, 2000.

[14] D. Gonzalez, A. George, and M. Chidester, "Performance Modeling and Evaluation of Topologies for Low-Latency SCI Systems," *Microprocessor and Microsystems*, Vol.25, No.7, pp. 343-356, 2001.

[15] R. Todd, M. Chidester, and A. George, "Comparative Performance Analysis of Directed Flow Control for Real-Time SCI," *Computer Networks*, Vol.37, No.4, pp. 391-406, 2001.

[16] H. Bugge, "Affordable Scalability using Multicubes," in: H. Hellwagner, A. Reinfeld (Eds.), SCI: Scalable Coherent Interface, LNCS State-of-the-Art Survey, Springer, Berlin, Germany, 1999, pp. 167-174.

[17] L.P. Huse, "Collective Communication on Dedicated Clusters Of Workstations," *Proc. of 6th PVM/MPI European Users Meeting (EuroPVM/MPI '99)*, Sabadell, Barcelona, Spain, Sep. 1999, pp. 469-476.

[18] S. Oral and A. George, "Multicast Performance Analysis for High-Speed Torus Networks," *Proc. of 27th IEEE Conference on Local Computer Networks (LCN), HSLN Workshop*, Tampa, FL, Nov. 2002, pp. 619-628.

[19] M. Gerla, P. Palnati, and S. Walton, "Multicasting Protocols for High-Speed, Wormhole-Routing Local Area Networks," *Proc. of SIGCOMM '96 Symp.*, Aug. 1996, pp. 184-193

[20] K. Verstoep, K. Landgendoen, and H. Bal, "Efficient Reliable Multicast on Myrinet," *Proc. of 1996 Int. Conf. On Parallel Processing*, Aug. 1996, pp. 156-165,

[21] P. Kesavan and D.K. Panda, "Optimal Multicast with Packetization and Network Interface Support," *Proc. of 1997 Int. Conf. on Parallel Processing*, Aug. 1997, pp. 370-377.

[22] R.A.F. Bhoedjang, T. Ruhl, and H.E. Bal, "Efficient Multicast on Myrinet Using Link-Level Flow Control," *Proc. of 1998 Int. Conf. On Parallel Processing*, Aug. 1998, pp. 381-389.

[23] D. Buntinas, D. K. Panda, and P. Sadayappan, "Fast NIC-Based Barrier over Myrinet/GM," *International Parallel and Distributed Processing Symposium, IPDPS'01*, pp. 52-60, San Francisco, CA, Apr. 2001.

[24] D. Buntinas, D. K. Panda, J. Duato, and P. Sadayappan, "Broadcast/Multicast over Myrinet using NIC-Assisted Multidestination Messages," *Proc. of Fourth Int. Workshop on Communication, Architecture, and Applications for Network-Based Parallel Computing, CANPC '00*, Toulouse, France, Jan. 2000, pp. 115-129.

[25] C. Kurmann and Thomas M. Stricker, "A Comparison of two Gigabit SAN/LAN technologies: Scalable Coherent Interface versus Myrinet," Scalable Coherent Interface: Technology and Applications Edited by Hermann Hellwagner & Alexander Reinefeld, pp. 29-42, Cheshire Henbury Publications, ISBN 1-901864-02-2, 1998.

[26] M. Fischer, U. Brüning, J. Kluge, L. Rzymianowicz, P. Schulz, and M. Waack, "ATOLL a new switched, high-speed Interconnect in Comparison to Myrinet and SCI," *IPDPS 2000 (International Parallel and Distributed Processing Symp.), PC NOW Workshop*, Cancun, Mexico, 2000

[27] H. Wang, and D.M. Blough, Tree-Based Multicast in Wormhole-Routed Torus Networks, *Proc. of Int. Conf. on Parallel and Distributed Processing Techniques and Applications (PDPTA'98)*, Las Vegas, Nevada, Jul. 1998, pp 702-709.