Topology and Routing in Clusters: From Theory to Practice

Yoav Etsion   Mickael Raizman   Dror G. Feitelson*

School of Computer Science and Engineering

The Hebrew University of Jerusalem

91904 Jerusalem, Israel

Abstract

Designers of communication subsystems for clusters often present performance data by measuring bandwidth and latency for single point-to-point connections. Such data is not sensitive to routing algorithms and to network topology, and little experimental evidence relating to the impact of these factors on performance has been collected. On the other hand there is abundant theoretical evidence for the use of topologies with high bisection bandwidth and multiple and even randomized routes. We show that indeed these ideas can be applied in practice to achieve significant benefits. In the case of topology, we show that the characteristics of commercially available Myrinet switches allow for the construction of related topologies with relatively few added links that provide much better support for intensive communication patterns. In the case of routing, we show simple mechanisms for implementing multiple paths in FM and randomization by randomized mapping of logical nodes. These mechanisms alleviate congestion due to uneven traffic patterns.

1 Introduction

Clusters of commodity PCs connected by fast LANs are becoming a common architecture for high-performance computing [25]. An interesting design issue for such clusters is how to configure the LAN for best performance. This includes the choice of topology, the routing algorithm, and the interplay between them. While these issues have received much attention in the professional literature, there is little experimental data regarding the match between theory and practice.

*feit@cs.huji.ac.il, tel: +972 2 658 4115, fax: +972 2 651 1912.

Two recurring ideas in interconnection network research are the use of topologies with a high bisection bandwidth, and the use of dynamic routing rather than a single predefined route between each pair of nodes. We investigated these ideas using the ParPar cluster as our experimental testbed. This is a cluster of 16 Pentium-Pro 200 PCs connected by a Myrinet LAN. Communication software is based on Fast Messages.

In our experiments we checked a set of related topologies with increasing bisection bandwidth. We found that indeed having a very sparse topology may hurt the achievable peak bandwidth. However, the bisection should not be measured in units of "number of links", but rather in units of bandwidth. In our case, the network is faster than the nodes, so fewer links than the "ideal" bisection are sufficient. We postulate that the opposite is also true: a relatively slow network can be compensated for by using more links, provided the routing mechanism knows how to use them.

In another set of experiments, we compared the use of a single route between each pair of nodes, using alternate routes, and using a randomized route. The results in this case were that using alternate routes improved performance, but randomization improved it even more. The implication is that a cluster management system need not take pains to map nodes in an orderly manner; on the contrary, mapping the logical nodes randomly to physical nodes may promote more efficient use of the network.

It should be noted that in all cases our test programs were *not* optimized for the topology and routing algorithm being used. This reflects common usage on clusters, which are perceived as off-the-shelf general-purpose platforms for high performance computing. For example, parallel programs on many clusters are based on using the MPICH implementation of the standard MPI interface [16]. This implementation is popular because it is portable: the same core algorithms are always used, and only a thin abstraction layer has to be rewritten for each new platform. Our research is aimed at finding mechanisms that may be expected to work fairly well for a broad range of applications, rather than at attempting to achieve optimal performance at the cost of significant specialization.

1.1 Myrinet

Our experimental platform is based on the Myrinet product from Myricom. Myrinet is a gigabit-per-second switched LAN [6], which is a leading candidate for high-performance communication in cluster environments [2]. Creating a Myrinet requires three components: a network interface card with memory and a communications co-processor (called the LANai) in each node, one or more switches, and cables to connect the nodes to the switches and the switches to each other.

In principle, any topology can be used. In practice, the usable topologies are constrained by the types of switches that are available. Until recently, the most common switch was the

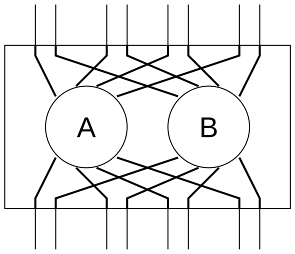

Figure 1: *Dual 8-way Myrinet switch: schematic and picture of switch with cables.*

dual 8-way switch, which is a box containing two switches with 8 ports each. The switches are denoted A and B. The box itself has 8 two-link ports, each serving a pair of switch ports, one from each switch (Fig. 1). Normally both links can be accommodated in a single cable, but a splitter can be used to separate the two links and channel them to different cables.

The best way to connect up to 8 nodes is to use only one of the two switches in the box. When using both, up to 14 nodes can be connected (because at least one port on each switch is needed in order to connect the switches to each other). Therefore at least two dual switches are needed in order to create a 16-node cluster. In such a configuration there are a total of 32 ports: 16 for the nodes, and another 16 for 8 inter-switch links, leading to a design such as that shown on the top of Fig. 2. The connectivity can be increased by using 3 dual switches, as shown at the bottom of the figure (actually this design can support 18 nodes).

For larger clusters, more switches are needed. Also, an oct switch can be used as the basic building block. This is a box containing eight 8-way switches organized in two layers of four switches each. One layer provides 16 singleton ports for connecting to nodes, and the other provides 8 two-link ports for connecting to other switches. Another alternative is the recent 16-way switch, which provides a building block that falls between an 8-way switch and an oct switch box.

While our work is based on specific configurations that can be created from Myrinet switches, the ideas explored are relevant for any cluster using a switched LAN. This includes implementations of the popular Fast Ethernet and Gigabit Ethernet LANs.

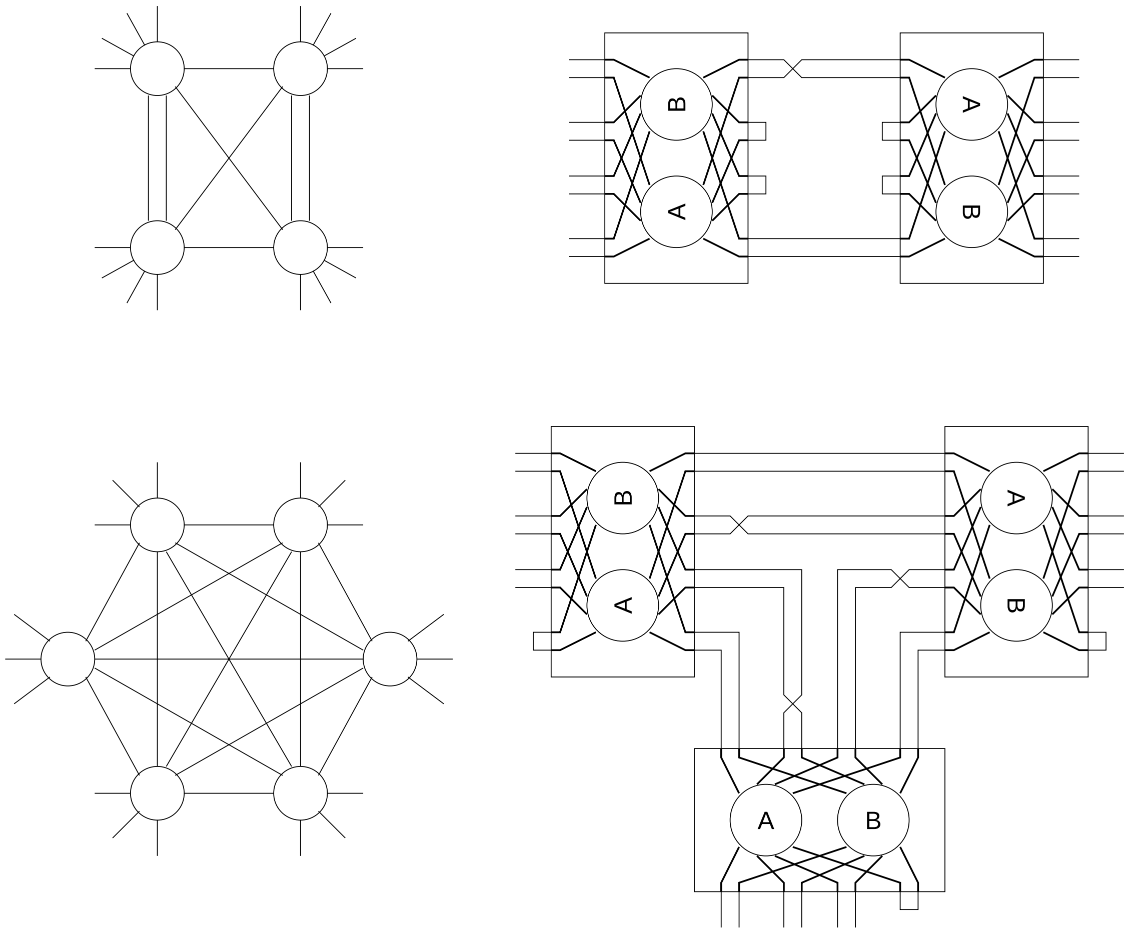

Figure 2: *Topologies for 16- and 18-node clusters with two or three dual switches. Left: the abstract topology. Right: the Myrinet implementation.*

1.2 Fast Messages

Fast Messages (FM) is a high-performance communication library for Myrinet, operating at the user-level (that is, sending messages does not involve the operating system kernel) [24]. To integrate FM with the ParPar system, the daemons used by FM to set up its environment (GRM and CM) were replaced by functions in the ParPar daemons. This was relatively straightforward, as the required data — such as job ID and logical node numbers — exists in the ParPar system anyway [14]. The only addition was an API that passes the data to the communication subsystem. The details are described elsewhere [13].

Routing in FM is static. Upon initialization, nodes read a configuration file that describes the network topology. They then compute minimal paths from each node to all the others, using Dijkstra's algorithm. If several minimal paths exist, the one using links with the least

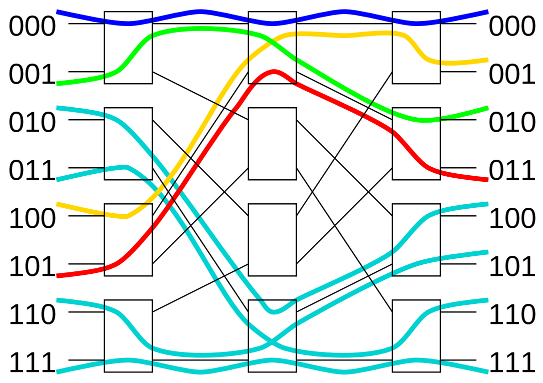

Figure 3: *Contention for switches and links as a result of a permutation on the shuffle-exchange network.*

usage by previous paths is chosen. The paths are encoded as a routing header with one byte per switch, and stored in a routing table on the LANai. When sending a message to node i, the LANai prepends the routing header for node i to the message and injects it into the network. Each switch along the path strips off one byte, performs the routing indicated by it, and forwards the rest of the message.

2 Network Contention

Contention for network resources has long been recognized as a major factor influencing performance. In this section we first outline some theoretical background that motivates our work, and then discuss some practical considerations in testing this theory.

2.1 Theory

Many regular networks suffer from contention when handling certain permutations of messages. Consider a shuffle-exchange multistage network connecting $N = 2^n$ nodes as an example. Such a network is composed of n stages of $N/2$ 2×2 switches, with switch i in one stage connected to switches $2i$ and $2i + 1 \bmod N/2$ in the next stage (Fig. 3). Routing is done by inspecting the destination address bit by bit, with each bit dictating the switching at one stage of the network: for 0 use the top link, and for 1 use the bottom one [20]. Now consider a traffic pattern where node i sends a packet to a destination node whose address is i cyclicly left-shifted by one bit. This is a permutation, so there is no contention for the nodes themselves. However, as the figure shows, the top and bottom switches in the middle stage become congested, while the middle two are idle.

The solution to this problem has been the subject of intense research, and has led to two

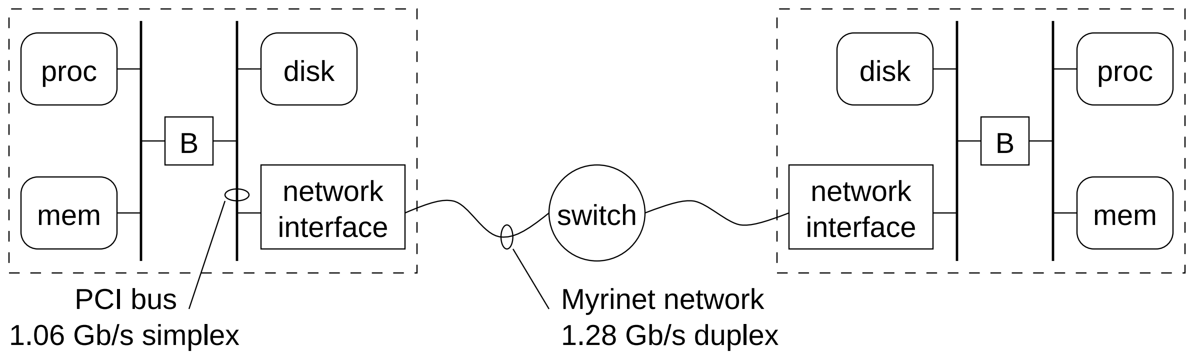

Figure 4: *Memory-to-memory communication path.*

complimentary notions. The first is the need for sufficient bisection bandwidth. The second is randomized routing.

Bisection bandwidth deals with the basic infrastructure for communication: switches and links. The idea is simple: if you don't have enough infrastructure to route the different communications using disjoint links, there is no way to escape from contention. The shuffle-exchange network described above does have sufficient bisection bandwidth; other network topologies such as a mesh do not.

However, having enough links does not help if the desired communication pattern does not use them, as shown above. This is where randomized routing helps: if messages are first routed to a randomly-chosen intermediate node, and then forwarded to their true destination, the structure and symmetry of the communication pattern are broken. When this happens the probability of congestion is extremely low [29, 21].

2.2 Implications for Test Programs

In testing the effect of network contention on bandwidth, we wish to operate in the domain where the network is the limiting factor. To do so we must identify the bandwidth constraints imposed by various parts of the system, and minimize communication overheads.

Fig. 4 shows a schematic diagram of the path traversed by data being sent from one computer to another. Once everything is set up, the processor copies the data from the memory to the network interface card, using the memory bus and the I/O bus (this programmed I/O is used for sending in FM; in principle, a DMA implementation is also possible). The data is then sent out on the network. In our test environment, the network is a Myrinet capable of peak throughput of 1.28 Gb/s in each direction. The bottleneck is the PCI bus, which is 32 bits wide and operates at 33 MHz, for a total peak throughput of 1.056 Gb/s. This peak bandwidth cannot be achieved due to overheads in handling the bus, and might be further reduced due to contention for the bus by disk accesses.

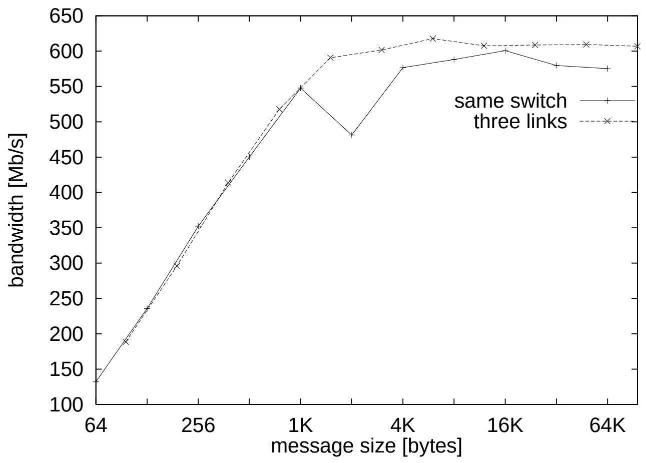

Figure 5: *Point-to-point bandwidth achieved by FM is essentially the same regardless of the number of Myrinet links traversed. The dip at 2K messages is due to fragmentation resulting from using packets of 1536 bytes. In the other graph all measurements are multiples of the packet size.*

Moreover, the software also has its overheads for packetization, flow control, etc. Our measurements indicate that the peak bandwidth achievable on our system (in a simple one-way point-to-point pattern) is only about 600 Mb/s, i.e. only 60% of the peak PCI bandwidth (Fig. 5). While higher peak bandwidths have been reported recently for FM [19], these relate to a newer optimized version of the software for an NT platform; our work is based on an older version, ported from a Linux platform.

The mismatch between the peak achievable bandwidth of about 0.6 Gb/s and the total two-way peak bandwidth of the Myrinet which is over 2.5 Gb/s is a cause for concern. It means that only communication patterns in which multiple connections contend for the same Myrinet link can hope to saturate the network. In addition, care should be taken to get the maximal bandwidth possible from each connection. To do so, we wrote our test programs in FM directly, rather than using a higher-level interface such as MPI, because the overhead of an additional software layer can reduce the peak achievable bandwidth by 10% or more [19]. Furthermore, we used a special crippled implementation of FM which discards incoming data rather than copying it to user-space buffers. This reduces software overhead and allows higher loading of the network.

2.3 Related Work

The adverse effects of congestion have been shown experimentally in a number of other studies. An interesting phenomenon is that in some cases reducing the congestion in an artificial manner improves overall performance. This is the basis for throttling mechanisms, which limit the load placed on the network so as not to overload it [8, 27, 28].

Indeed, the balance between processing power and network bandwidth has been identified as an important design goal. Recently, Becker et al. have shown that this consideration indicates that small clusters based on the Pentium Pro processor (like ParPar) can make do with a fast Ethernet, and do not require a Myrinet; only very large clusters justify the more expensive Myrinet [5]. This can be related to earlier results by Boothe and Ranade, which claimed that the bandwidth required is 1–2 bits per op [7]. Our results corroborate these conclusions.

Three classical topologies (fully connected, mesh, and hypercube) were compared in [26], for use with a set of specific applications. This was then used to gauge the bandwidth requirements of these applications. However, such data is not directly useful for the design of general-purpose clusters.

The contribution of our work is to investigate the questions of topology and routing experimentally in a realistic setting, with the goal of providing guidance for the construction of clusters. Thus our topologies reflect real packaging and configuration constraints, rather than being abstract ideals. However, we do check notions that have their roots in theory, such as bisection bandwidth and randomized routing.

3 Measurement Methodology

As noted above, we need to use communication patterns in which multiple connections traverse a single link in order to saturate it. In other words, our communication patterns will have many senders and many receivers. The question is then how to measure performance.

In a nutshell, our performance metric is the total bandwidth delivered by the network. This is defined to be the total number of bits transferred divided by the time needed to complete all the transfers in the communication pattern. The total number of bits is the sum over all sender-destination pairs of the number of bits sent from that sender to that destination. The completion time is the difference between the start time of the first process involved and the end time of the last one to finish (similar to [23]).

While this definition may sound rather straightforward, there are some subtleties. Consider a broadcast operation, in which one node sends the same b bits to k other nodes. In our formulation, this is measured as $k \cdot b$ bits, regardless of how many bits were actually sent: it could be that a broadcast medium was used, and therefore only one instance of b bits was actually sent, or it could be that some data was forwarded from one node to another, and therefore a total of more than $k \cdot b$ bits were transmitted. The point is that these are implementation details, and we are interested in the logical equivalent of the communication operation.

Technically, the measurements are performed as follows: first, all nodes perform a barrier synchronization. They then execute the desired communication pattern repeatedly for a large number of times. When they finish, they perform another barrier synchronization. The time measured is from after the first barrier to after the second one. The number of iterations is chosen to be large enough so that the overhead of the barriers themselves is negligible. The barriers are implemented as an all-to-one pattern followed by a one-to-all pattern, i.e. node 0 serves to coordinate all the others. We also checked the individual times measured *between* the barriers by the different processors, and found that the differences among these values and the value measured including the barriers is less than 1%. This means that generally all the nodes were really active all the time.

Our experiments revealed that another problem is context switching in the LANai processor on the network interface cards. This processor can only handle one direction of data flow at a time, and gives priority to incoming messages. Thus if it senses an incoming message while it is busy sending an outgoing message, the sending is stopped, and the incoming message is handled instead. When it ends, the sending of the outgoing message is resumed. Such switching causes overhead on the LANai and reduces the peak bandwidth. We therefore designed additional test programs to alternate between sending and receiving, rather than to attempt to do both at once. These were compared with the simple version.

4 Routing Using Multiple and Random Paths

The FM routing algorithm finds a shortest path between each pair of nodes, and if there are several shortest paths, chooses the one that shares its links with fewer other routes. To find two disjoint paths among each pair of nodes, we first run this algorithm to find one path as before. Then we set the usage of the links of this path to the maximal possible usage, effectively eliminating them from consideration. We then run the algorithm again, and find another (possibly non-minimal) route. The fact that the alternative route is not minimal does not affect its performance, as forwarding the messages trough multiple Myrinet switches has practically no effect (Fig. 5).

In order to use both routes, we double the size of the FM routing table that is stored on the LANai. Each destination node is then represented by two entries in the enlarged table, separated by the cluster size. For example, in a 16-node cluster, routes to node 0 are stored in entries 0 and 16, routes to node 1 in entries 1 and 17, and so on. The FM send routine chooses which route to use, and forwards the matching alias to the LANai. The LANai itself is not cognizant of the fact that pairs of routes actually refer to the same node. While

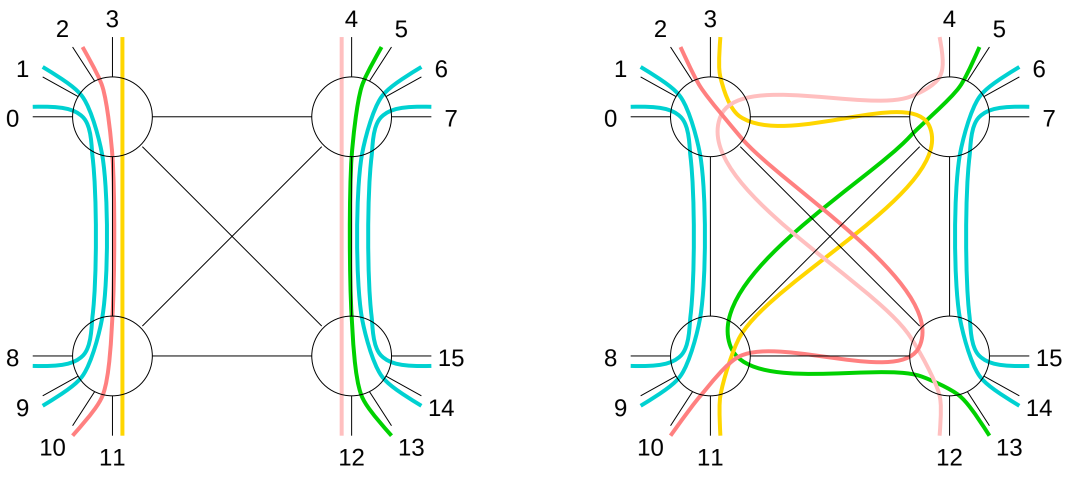

Figure 6: *Communication patterns for pairs of nodes using only minimal routes (left) and alternative 2-hop links (right).*

memory on the network interface card is scarce (our cards have 512 KB), a routing table for a 16 node cluster is only 64 bytes, so doubling its size is still negligible.

There are at least three ways in which to choose a route:

- Use both routes alternatively. This method may benefit from spreading the traffic more evenly across the network. In order to implement it, the send routine must keep a 1-bit flag for each destination node, indicating the route to be used next time. Sending to node i is then performed by the following lines of code:

  ```
  d = i + flag[i] * SIZE;
  flag[i] = 1 - flag[i];
  ```

  The first line adds the cluster size to the original address if the flag bit is set, and the second toggles the flag bit.

- Make a random choice. The send routine calls a random bit generator, and uses its output to decide which route to use.

- Use the least loaded route. While this option seems very promising and may also provide a measure of fault tolerance, it is hard to implement because it requires precise on-line measurement of performance. Indeed, previous work in this direction has been limited to choosing among different types of networks based on static characteristics of the message (e.g. one network for short messages and another for long ones) [18].

To check the effect of having multiple paths, we chose a communication pattern in which pairs of nodes communicate among themselves. In our sample 16-node topology, if the pairs

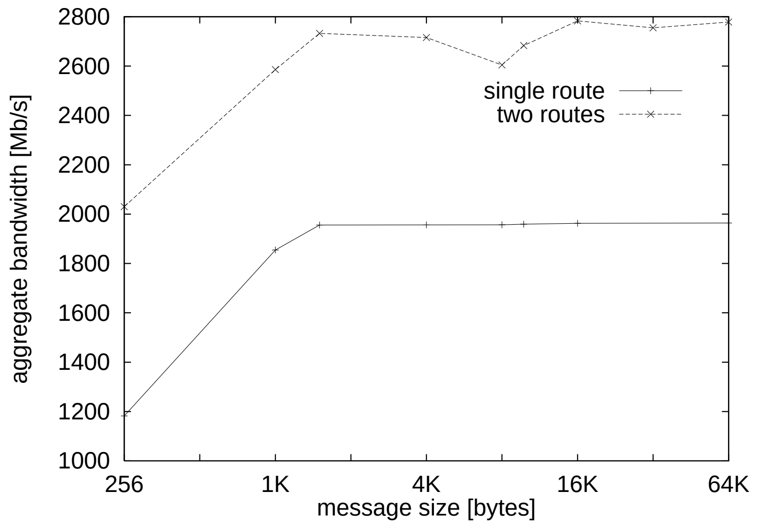

Figure 7: *Aggregate bandwidth as function of message size when using a single minimal path or two paths alternatively.*

are composed of nodes whose IDs differ by 8, this leads to two links that carry 4 connections each, while other links are free (Fig. 6 left). However, when 2-hop alternative paths are allowed, it is possible to utilize all the links and reduce the maximal load on any single link to only 2 connections (right of figure). While the FM routing mechanism will not necessarily choose exactly these routes, this pattern shows the potential for improvement.

The measurement results are shown in Fig. 7. Over the complete range of message sizes, using two routes alternatively improves the aggregate achieved bandwidth by about 30%. The peak aggregate bandwidth that was achieved was 1.96 Gb/s for the original routing, and 2.78 Gb/s for the two-route routing. As the original routing uses only two Myrinet links, and the pattern is completely symmetrical, it is reasonable to assume that these two links each carry traffic at 0.98 Gb/s. Thus even combining four data streams of 0.6 Gb/s each does not lead to full utilization of the links, but suffers from some inefficiencies.

Another idea we wanted to check was the use of randomized routing. Rather than randomize the routing itself, we randomized the mapping of logical node numbers to physical processing elements. As shown in Fig. 8, this can be expected to reduce the contention for regular communication patterns.

The results were an additional dramatic improvement in the total bandwidth delivered by the network, which now peaked at 4.53 Gb/s (Fig. 9). This is partly due to the fact that some of the pairs of communicating nodes get mapped to the same switch, thus leading to a considerable reduction in the contention for network links. As may be expected, the use of two alternate routes did not make a very big difference in this setting. However there is a

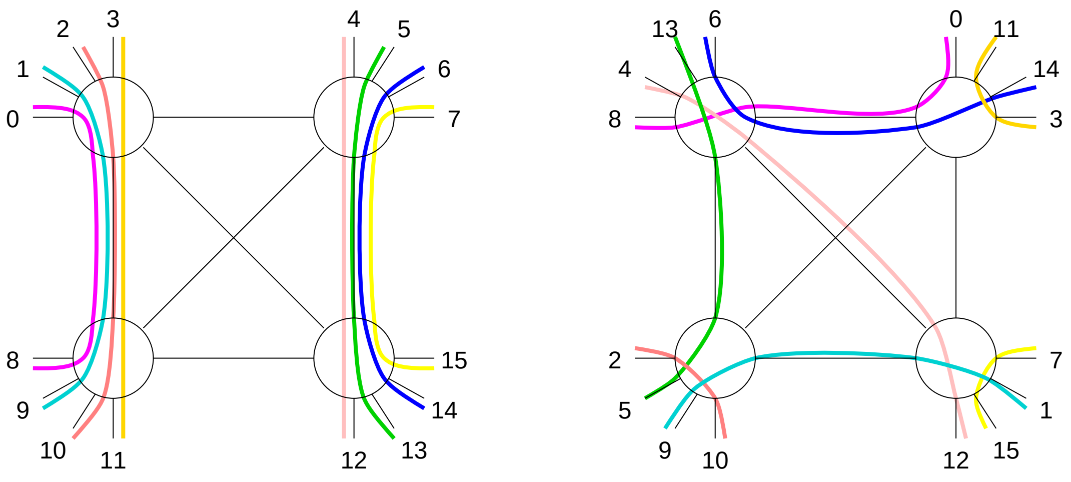

Figure 8: *Communication patterns for pairs of nodes using minimal routes, when nodes are ordered (left) or randomized (right).*

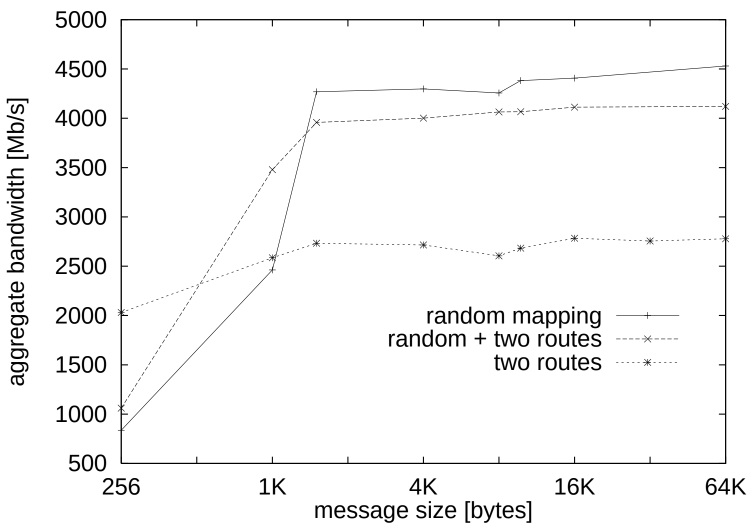

Figure 9: *Aggregate bandwidth as function of message size when the mapping of logical nodes is randomized. The plot for using two alternate routes from Fig. 7 is shown for comparison.*

definite difference, and the use of two routes actually reduces the performance by about 8%. The reason is that in the cases where two communicating nodes happen to be connected to the same switch, forcing them to use an alternative route creates a needless loop that imposes extra load on other switches and links.

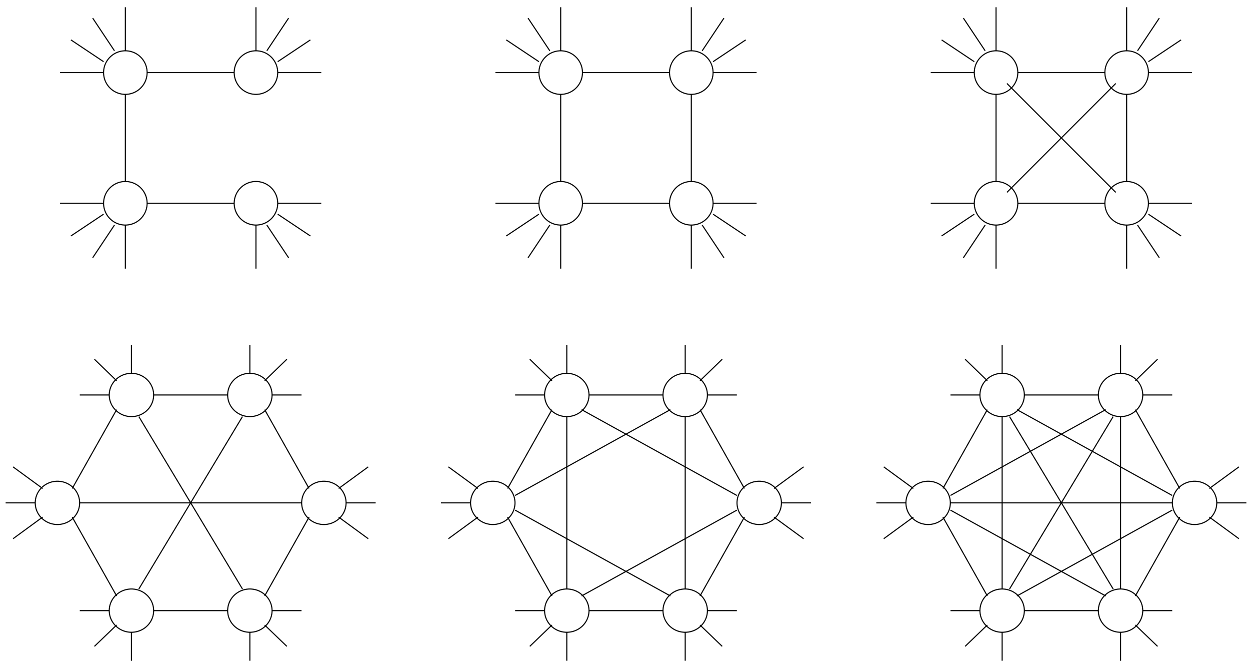

Figure 10: *Topologies for Myrinet-based clusters of 16 nodes with bisections of 1, 2, 4, 5, 6, and 9 links.*

5 Bisection Bandwidth

Bisection bandwidth is one of the metrics by which network topologies are measured. A general requirement is that a network of N nodes have a bisection bandwidth of $O(N)$, meaning that when all nodes are sending messages to each other the contention will cause no more than a constant delay. For example, a mesh has a bisection of $\sqrt{N}$ links, which is too small, whereas a hypercube has a bisection of $N/2$ links, which is just right.

It is most common to consider network topologies as abstract graphs, and measure bisection bandwidth as the minimal number of edges that have to be removed in order to partition the graph into two equal parts. An alternative approach is to take realistic packaging constraints into consideration, and investigate the effect of limitations on achievable bisection bandwidth on performance [17, 12]. The switch configurations sold by Myricom also limit the types of networks that can be constructed. For example, Fig. 10 shows six alternative topologies that can be used to create a 16-node network, with increasing bisection bandwidth. The first three use two dual switches, leading to a maximal bisection bandwidth of 4, which falls short of the desired value of 8; using three switches it is possible to reach a bisection bandwidth of 9.

Obviously creating a network with a higher bisection bandwidth is just a question of acquiring additional switches and cables. The question a cluster designer must pose is whether

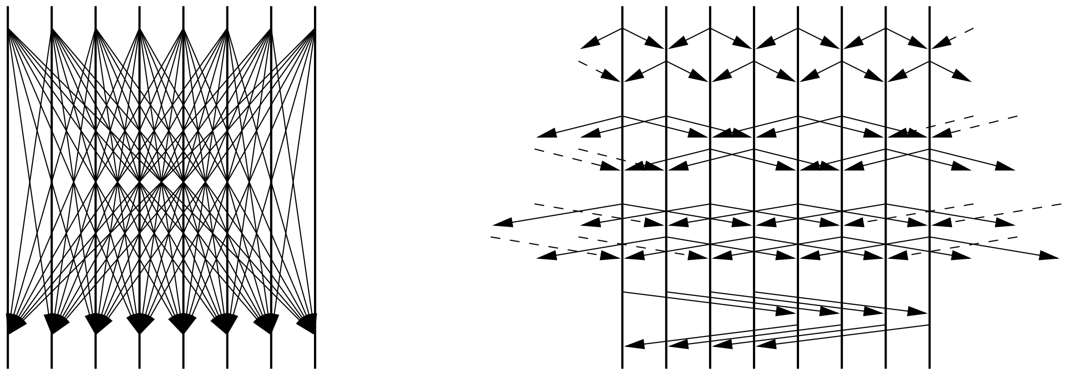

Figure 11: *Alternative implementation of the all-to-all communication pattern, without and with phases.*

this investment is worth while. To investigate this issue, we measured the communication performance of the topologies shown in Fig. 10, using an all-to-all communication pattern. These topologies were selected — in addition to the range of bisection bandwidths they possess — because they are all simple modifications of the same two basic structures. This eases the configuration management and reduces the risk of wiring errors.

The all-to-all communication pattern was implemented in two alternative ways. In the first, each node first sends its messages to all other nodes, and then accepts any massages coming in to it (11 left). There is no deadlock because FM will automatically stop sending and extract data from the network if it runs out of buffers. In the second organization, nodes are paired so that each node sends and receives data in alternating phases of the algorithm (11 right). This is intended to reduce context switching on the LANai.

The experimental results are shown in Fig. 12. Obviously the network's bisection bandwidth has an effect on the achievable total bandwidth for this communication pattern. An interesting observation is that for the lowest bisection bandwidth, i.e. the case where there is the most contention, the more organized implementation of all-to-all communication with phases provides the better performance. For higher bisection bandwidths, the unorganized implementation is better. Moreover, the organized pattern is much more stable for large message sizes. The unorganized one suffers a degradation in performance when large messages are involved, due to uncontrolled contention.

To assess the importance of network bisection, this data is shown again in Fig. 13. Five different communication patterns are studied: the two versions of all-to-all communication used in Fig. 12, two versions of one-to-all (broadcast) communication, one using a serial loop on all destinations and the other using a spanning tree, and an all-to-one (reduce) pattern. In all five patterns it is clear that the message size has a strong effect on achievable bandwidth, with messages of at least one packet size required (a packet in FM is 1536 bytes). The

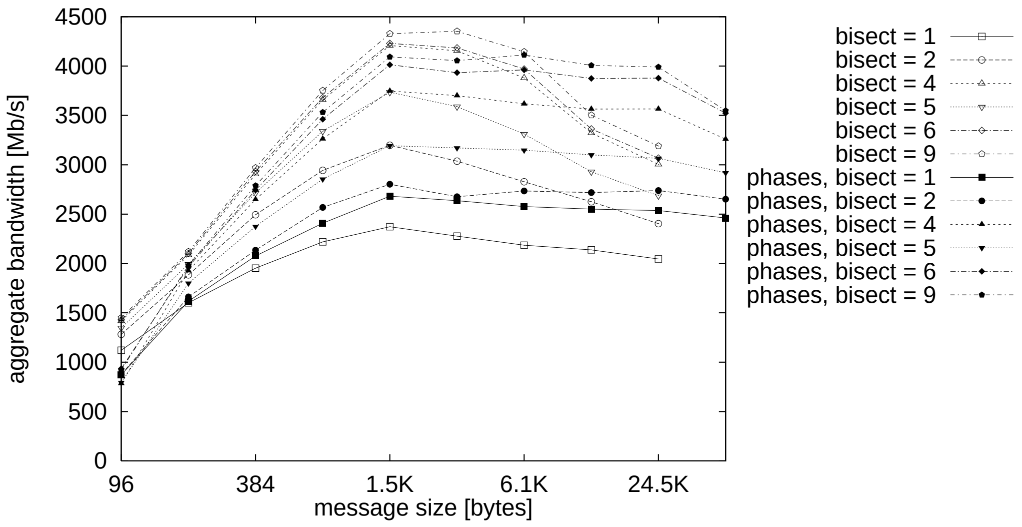

Figure 12: *Bandwidth as a function of message size for all-to-all communication and different bisection bandwidths.*

bisection also has an effect, but it is much smaller.

Fig. 14 shows the results from a different angle: peak bandwidth as a function of bisection bandwidth, using the message size that achieves the best performance in each case. Again all five communication patterns are included. As may be expected, the all-to-one and serial one-to-all patterns do not benefit from increasing the bisection bandwidth of the network, as they do not contain parallel activation of different connections. The all-to-one pattern seems to achieve a higher bandwidth, but this is actually a result of the fact that we discard incoming messages to reduce copying overhead. The one-to-all tree pattern benefits to a small degree from increased bisection, but only for the largest message size.

As for the all-to-all pattern, it seems to saturate and achieves essentially the same bandwidth for topologies with a bisection of 4 or more. This is due to the fact that indeed the topology with bisection 4 provides enough bandwidth, and adding more does not help. This indicates that counting bisection in links is wrong — it should be counted in bandwidth. In a configuration like that of our test platform, where the network is overpowered, less links provide sufficient bandwidth.

6 Discussion and Conclusions

The most common approach to achieving high performance communication is to tailor the communication patterns and algorithms to the properties of the underlying network. This is the basis for programming models like LogP and its derivatives [11, 1, 15, 22] and the Postal

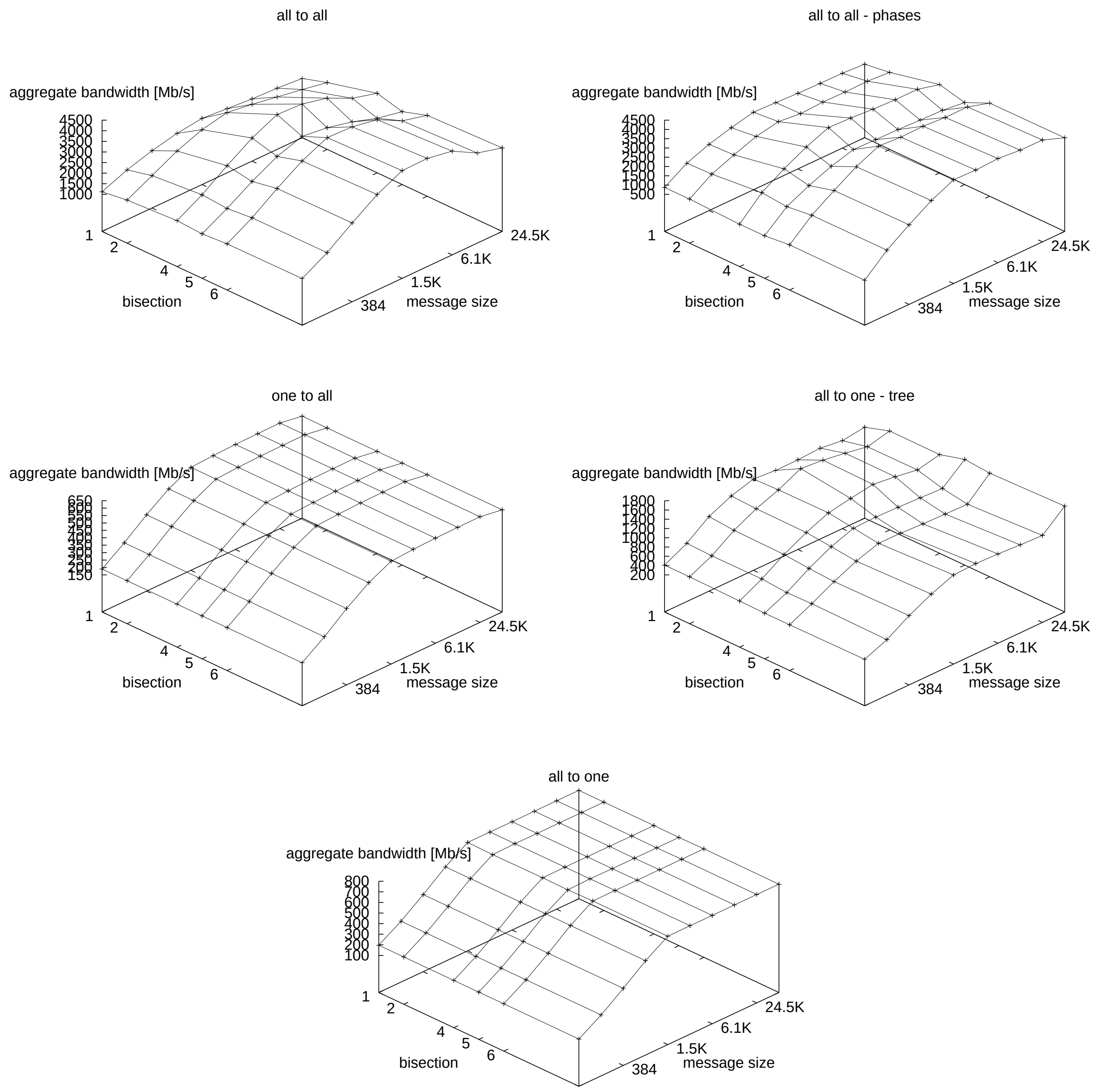

Figure 13: *Bandwidth as a function of bisection and message size for different communication patterns.*

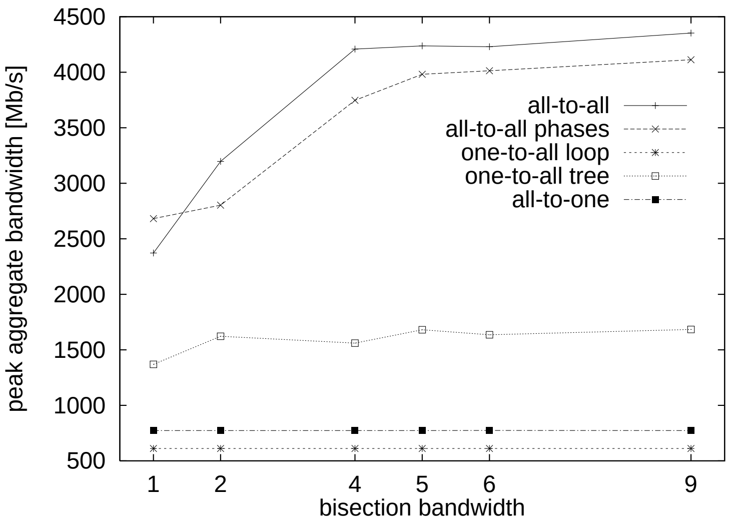

Figure 14: *Bandwidth as a function of bisection bandwidth for different communication patterns.*

component	*price*
PCI-network interface	$995.-
dual 8-way switch	$2000.-
3-foot cable	$120.-

Table 1: *List prices of Myrinet components as of November 1999.*

model [9], and for numerous algorithms for the implementation of collective communication primitives [10, 3, 4]. In contrast, we focus on *general mechanisms that do not require detailed knowledge of network properties.*

Our work was inspired by well-known theoretical work promoting ideas such as randomized routing, using multiple paths, and creating topologies with a high bisection bandwidth. We showed a simple implementation of multi-route routing in the context of the FM communication system. Of special interest is the demonstration that using alternative non-minimal routes may lead to significant performance improvements. The common wisdom is that minimal routes should be used, because using non-minimal routes presumably increases the total load on the network and the danger of congestion.

We also showed that using a randomized mapping of logical nodes is a good alternative to randomized routing, and breaks bad traffic patterns, leading to much improved performance. These mechanisms are simple enough to be added easily to the system software of any cluster environment, and will be useful even when ported to other configurations or platforms.

With regard to bisection bandwidth, we showed that for intensive communication patterns there is a significant difference between the achieved performance on a minimal network with low bisection bandwidth, and a more lavish one with higher bisection bandwidth. An important observation is that this may involve only a very small difference in cost. For example, given the current price of Myrinet components (Table 1), the list price of our system with a network bisection of 1 is $22,200. The difference for constructing a system with network bisection of 4 is just adding 3 cables, for a price of $360, or 1.6%. The difference for a system with bisection of 9 is one additional dual switch and 12 cables, for a total of $3440 or 15.5%. This is actually not needed, as we found that a network with bisection 9 has hardly any benefit over one with bisection 4. But in either case, the network interface cards dominate the price; adding bisection bandwidth is relatively cheap.

While we have demonstrated that using the ideas of multiple routes, randomized routing, and large bisection bandwidth can lead to significant performance gains at low cost and complexity, much remains to be done. A major concern is improving our understanding of the performance of general implementations of collective communication. Our measurements show that the peak performance of collective communications represents only a low utilization of the links. For example, the peak all-to-all bandwidth measured on the bisection 4 topology was 4.2 Gb/s. As this topology is completely symmetric, and so is the communication pattern, it is reasonable to postulate that all 6 links contribute equally to this result. This leads to an estimate of about 0.7 Gb/s on each link, which is only 27.5% of the two-way capacity of 2.5 Gb/s. The fact that point-to-point communications achieve 47% of the one-way capacity leads us to believe that the problem is with contention in the switches. The challange is to design *general* mechanisms that alleviate this problem, rather than resorting to specialized algorithms for each communication pattern.

Acknowledgements

This research was suported in part by the Israel Science Foundation founded by the Israel Academy of Sciences & Humanities. We are grateful to Prof. Andrew Chien for providing us with a source license for FM.

References

[1] A. Alexandrov, M. F. Ionescu, K. E. Schauser, and C. Scheiman, "*LogGP: incorporating long messages into the LogP model — one step closer towards a realistic model for parallel computations*". In 7th *Symp. Parallel Algorithms & Architectures*, pp. 95–105, Jul 1995.

[2] H. Bal, R. Hofman, and K. Verstoep, "*A comparison of three high speed networks for parallel cluster computing*". In *Communication and Architectural Support for Network-Based Parallel*

Computing, D. K. Panda and C. B. Stunkel (eds.), pp. 184–197, Springer-Verlag, Feb 1997. Lect. Notes Comput. Sci. vol. 1199.

[3] V. Bala, J. Bruck, R. Cypher, P. Elustondo, A. Ho, C-T. Ho, S. Kipnis, and M. Snir, "*CCL: a portable and tunable collective communications library for scalable parallel computers*". *IEEE Trans. Parallel & Distributed Syst.* **6(2)**, pp. 154–164, Feb 1995.

[4] M. Barnett, S. Gupta, D. G. Payne, L. Shuler, R. van de Geijn, and J. Watts, "*Interprocessor collective communication library (InterCom)*". In *Scalable High-Performance Comput. Conf.*, pp. 357–364, May 1994.

[5] J. C. Becker, B. Nitzberg, and R. F. van der Wijngaart, "*Predicting price/performance trade-offs for Whitney: a commodity computing cluster*". *J. Supercomput.* **13(3)**, pp. 303–319, May 1999.

[6] N. J. Boden, D. Cohen, R. E. Felderman, A. E. Kulawik, C. L. Seitz, J. N. Seizovic, and W-K. Su, "*Myrinet: a gigabit-per-second local area network*". *IEEE Micro* **15(1)**, pp. 29–36, Feb 1995.

[7] B. Boothe and A. Ranade, "*Performance on a bandwidth constrained network: how much bandwidth do we need?*". In *Supercomputing '93*, pp. 906–915, Nov 1993.

[8] E. A. Brewer and B. C. Kuszmaul, "*How to get good performance from the CM-5 data network*". In 8th *Intl. Parallel Processing Symp.*, pp. 858–867, Apr 1994.

[9] J. Bruck, L. De Coster, N. Dewulf, C-T. Ho, and R. Lauwereins, "*On the design and implementation of broadcast and global combine operations using the postal model*". *IEEE Trans. Parallel & Distributed Syst.* **7(3)**, pp. 256–265, Mar 1996.

[10] J. Bruck, C-T. Ho, S. Kipnis, and D. Weathersby, "*Efficient algorithms for all-to-all communications in multi-port message-passing systems*". In 6th *Symp. Parallel Algorithms & Architectures*, pp. 298–309, Jun 1994.

[11] D. E. Culler, R. M. Karp, D. Patterson, A. Sahay, E. E. Santos, K. E. Schauser, R. Subramonian, and T. von Eicken, "*LogP: a practical model of parallel computation*". *Comm. ACM* **39(11)**, pp. 78–85, Nov 1996.

[12] W. Dally, "*Network and processor architecture for message-driven computers*". In *VLSI and Parallel Computation*, R. Suaya and G. Britwistle (eds.), chap. 3, Morgan Kaufmann Publishers, Inc., 1990.

[13] Y. Etsion and D. G. Feitelson, *Integration of Gang Scheduling and User-Level Communication.* Technical Report in preparation, Hebrew University, 1999.

[14] D. G. Feitelson, A. Batat, G. Benhanokh, D. Er-El, Y. Etsion, A. Kavas, T. Klainer, U. Lublin, and M. A. Volovic, "*The ParPar system: a software MPP*". In *High Performance Cluster Computing, Vol. 1: Architectures and Systems*, R. Buyya (ed.), pp. 754–770, Prentice-Hall, 1999.

[15] M. I. Frank, A. Agarwal, and M. K. Vernon, "*LoPC: modeling contention in parallel algorithms*". In 6th *Symp. Principles & Practice of Parallel Programming*, pp. 276–287, Jun 1997.

[16] W. Gropp, E. Lusk, N. Doss, and A. Skjellum, "*A high-performance, portable implementation of the MPI message passing interface standard*". *Parallel Computing* **22(6)**, pp. 789–828, Sep 1996.

[17] A. C. Hartmann and J. D. Ullman, *Model Categories for Theories of Parallel Systems.* Technical Report PP-341-86, MCC, Austin, Texas, 1986. Reprinted in G. J. Lipovski and M. Malek, *Parallel Computing: Theory and Comparisons*, John Wiley & Sons, 1987, appendix H.

[18] J. Kim and D. J. Lilja, "*Performance-based path determination for interprocessor communication in distributed computing systems*". *IEEE Trans. Parallel & Distributed Syst.* **10(3)**, pp. 316–327, Mar 1999.

[19] M. Lauria, S. Pakin, and A. Chien, "*Efficient layering for high speed communication: the MPI over Fast Messages (FM) experience*". *Cluster Comput.* **2(2)**, pp. 107–116, Sep 1999.

[20] D. H. Lawrie, "*Access and alignment of data in an array processor*". *IEEE Trans. Comput.* **C-24(12)**, pp. 1145–1155, Dec 1975.

[21] F. T. Leighton and B. M. Maggs, "*Fast algorithms for routing around faults in multibutterflies and randomly-wired splitter networks*". *IEEE Trans. Comput.* **41(5)**, pp. 578–587, May 1992.

[22] C. A. Moritz and M. I. Frank, "*LoGPC: modeling network contention in message-passing programs*". In *SIGMETRICS Conf. Measurement & Modeling of Comput. Syst.*, pp. 254–263, Jun 1998.

[23] N. Nupairoj and L. M. Ni, "*Performance metrics and measurement techniques of collective communication services*". In *Communication and Architectural Support for Network-Based Parallel Computing*, D. K. Panda and C. B. Stunkel (eds.), pp. 212–226, Springer-Verlag, Feb 1997. Lect. Notes Comput. Sci. vol. 1199.

[24] S. Pakin, V. Karamcheti, and A. A. Chien, "*Fast messages: efficient, portable communication for workstation clusters and MPPs*". *IEEE Concurrency* **5(2)**, pp. 60–73, Apr-Jun 1997.

[25] G. F. Pfister, "*Clusters of computers for commercial processing: the invisible architecture*". *IEEE Parallel & Distributed Technology* **4(3)**, pp. 12–14, Fall 1996.

[26] A. Sivasubramaniam, A. Singla, U. Ramachandran, and H. Venkateswaran, "*An application-driven study of parallel system overheads and network bandwidth requirements*". *IEEE Trans. Parallel & Distributed Syst.* **10(3)**, pp. 193–210, Mar 1999.

[27] J. Stamatopoulos and J. A. Solworth, "*Increasing network bandwidth on meshes*". In 6th *Symp. Parallel Algorithms & Architectures*, pp. 336–345, Jun 1994.

[28] S. W. Turner and A. V. Veidenbaum, "*Scalability of the Cedar system*". In *Supercomputing '94*, pp. 247–254, Nov 1994.

[29] L. G. Valiant, "*A scheme for fast parallel communication*". *SIAM J. Comput.* **11(2)**, pp. 350–361, May 1982.