An Empirical Comparison of Area-Universal and Other Parallel Computing Networks

Ronald I. Greenberg
Mathematical and Computer Sciences
Loyola University
6525 North Sheridan Road
Chicago, IL 60626

Lee Guan
Electrical Engineering
University of Maryland
College Park, MD 20742

Abstract

This paper provides empirical comparison of the communication capabilities of two area-universal networks, the fat-tree and the fat-pyramid, to the popular mesh and hypercube networks for parallel computation. While area-universal networks have been proven capable of simulating, with modest slowdown, any computation of any other network of comparable area, prior work has generally left open the question of how area-universal networks compare to other networks in practice. Comparisons are performed using techniques of throughput and latency analysis that have previously been applied to k-ary n-cube networks and using various existing models to equate the hardware cost of the networks being compared. The increasingly popular *wormhole* routing model is used throughout.

keywords: performance evaluation, fat-tree, fat-pyramid, mesh, hypercube, area-universal networks, parallel computation

1 Introduction

Among the major impediments to massively parallel computing has been the difficulty of building a rich interconnection network and providing rapid communication among processors in a large system. Several works geared toward providing a general-purpose interconnection scheme with modest hardware cost have performed theoretical analyses of variants of the *fat-tree* architecture [3, 4, 8, 9, 11, 14, 15]. The theoretical results show that under a variety of routing models, a fat-tree is *area-universal*, i.e., it requires only a slowdown polylogarithmic in its area to simulate the behavior of any other network occupying the same area. This research has influenced the design of actual parallel computers by Thinking Machines Corporation and Meiko [5, 13, 16]. Little empirical evaluation of the practical performance comparison between area-universal networks and other popular networks for building large systems has been reported, however. This paper performs such a comparison under various standard models for equalizing hardware costs of the network realizations being compared. Empirical comparison is particularly interesting, since some empirical simulations of message routing on fat-trees have already shown better performance in practice than the performance predicted by provable bounds [11].

Throughout this paper, we use *wormhole* routing [7], due to its practical tendency to reduce routing time as well as the storage requirements of intermediate nodes. In this model, packets are composed of *flits* or *flow control digits*, and packets snake through the network one flit after another; only a constant number of flits may be stored in an intermediate node at any time.

The networks considered in this paper include the two extremes of the k-ary n-cube family considered by Dally [6], namely the mesh[1] and the (binary) hypercube, both of which are popular networks for parallel computers. We also consider the variant of the fat-tree referred to as the butterfly fat-tree, introduced by Greenberg and Leiserson [9]. Finally, we consider the fat-pyramid network [8], an augmentation of the fat-tree with hierarchical mesh connections, which was introduced to maintain area-universality without the usual simplifying assumption that unit time suffices to traverse any wire regardless of its length.

Different physical constraints have been used as measures of network cost. Dally [6] compared networks under the constraint that they have the same bisection width (minimum number of wires cut when the network is divided into two equal halves). The channel

[1] Actually, our mesh differs slightly from Dally's torus with wraparound connections.

widths (number of wires between adjacent nodes) of networks built on the same number of processors are adjusted to equalize bisection width. This comparison favored low-dimensional networks (closer to the mesh than the hypercube). Abraham and Padmanabhan [1] used the constraint of constant pin-out, the total number of wires emanating from the processors and switches in the network. There comparison (with some difference in the routing model) found higher dimensionality to be more important than wider channel width. In this paper, we expand the comparison to include area-universal networks using both of these simple constraints. We also compare the networks based on equal area in the Thompson model for VLSI [18, 19], the model that motivated the original design of area-universal networks. (Bisection width is closely related to but not the same as measuring area; Thompson showed that area is lower bounded by a constant times the square of bisection width.)

The rest of this paper is organized as follows. Section 2 describes the networks and routing model considered. Section 3 describes the constraints used to model hardware cost and obtain performance comparisons of networks of comparable cost. Section 4 gives the simulation results, and section 5 contains concluding remarks.

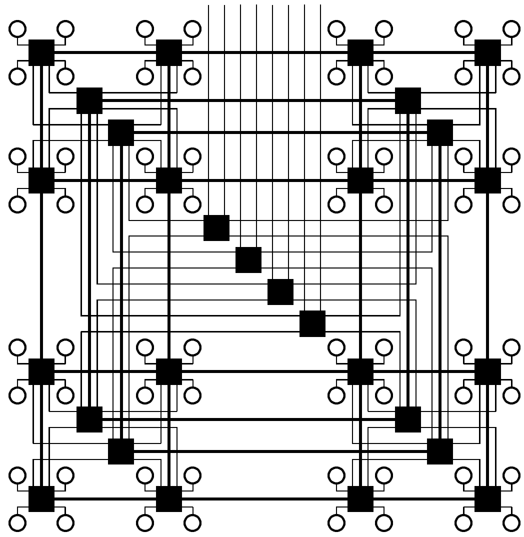

Figure 1: A fat-pyramid. Processors are placed at the leaves, represented by circles; the squares are switches. This network is obtained by superposing hierarchical mesh connections on a butterfly fat-tree. The original fat-tree connections are represented by thin lines and the mesh connections by thick lines. (A different layout of the fat-pyramid is used to obtain results independent of wire delay.)

2 Networks and Routing Model

For completeness, we begin by reviewing the interconnection pattern for the mesh and hypercube. Labeling the n processors of the mesh as (x, y) for $0 \le x, y < \sqrt{n}$, we connect (with bidirectional links) (x, y) to $(x+1, y)$ for $0 \le x < \sqrt{n} - 1$ and connect (x, y) to $(x, y+1)$ for $0 \le y < \sqrt{n} - 1$. Labeling the processors of the hypercube with the binary representations of the numbers 0 to $n-1$, two processors are connected if their labels differ in exactly one bit position (thought of as a physical dimension).

The structure of the fat-pyramid is shown in Figure 1 from [8]; the (butterfly) fat-tree is a subnetwork of the fat-pyramid. (We consider simple versions of these networks, though processors can be packed a bit more densely [8, Section II].) A precise description of the interconnection pattern for the switches in the fat-pyramid can be given as follows. We begin with a collection of ordinary two-dimensional meshes at levels $0, 1, \ldots, \log_2 \sqrt{n/4}$ representing distance from the leaves in the underlying tree structure. At level h, there are 2^h copies of a $\sqrt{n/4}/2^h \times \sqrt{n/4}/2^h$ mesh. We then denote a switch by an ordered 4-tuple (h, c, x, y), where h is the level, c is the copy number of the mesh ($0 \le c < 2^h$) at this level that contains the switch, and x and y specify a mesh position in an ordinary Cartesian coordinate system ($0 \le x, y < \sqrt{n/4}/2^h$). Then for $0 \le h < \log_2 \sqrt{n/4}$, switch (h, c, x, y) is connected to $(h+1, 2c, \lfloor x/2 \rfloor, \lfloor y/2 \rfloor)$ and $(h+1, 2c+1, \lfloor x/2 \rfloor, \lfloor y/2 \rfloor)$. The fat-tree is as above, but without the mesh connections.

In wormhole routing, messages (worms) are composed of *flits* (flow control digits). A worm snakes through the network one flit after another. An intermediate switching node starts to forward a worm to the next node (if there is no contention at the outgoing buffer to the next node) as soon as it has received enough flits to determine the destination of the worm. The switching nodes has a constant sized queue for each port so that only constant number of flits may be stored in each queue. When wormhole routing is used deadlock may occur. For the hypercube, *e-cube* routing [7] may be used to avoid deadlock. In e-cube routing, worms route through dimensions in the order specified by scanning the address bits from MSB to LSB. For a mesh (with no wraparound links as opposed to the related torus), it also suffices to route first in one dimension and then in the other. This deadlock avoidance strategy also applies to the hierarchical mesh networks in the fat-pyramid network.

Simulation programs have been developed to conduct the comparison of the four types of networks under wormhole routing. We focus on the unit delay model, in which the time to send a bit across a wire is independent of the wire length. In this context, simulation of the fat-pyramid shows the degree of performance loss resulting from using the more general network under the unit delay model; further work is planned to explore performance with differing wire delay. All comparisons are performed among networks that are expected to have comparable hardware cost, by adjusting the number of wires connecting nodes that are adjacent in the network as described in the next section.

3 The Constraints on Channel Width

A primary concern for performance comparison is the constraint on the channel width, the number of wires connecting adjacent nodes, to make the cost of interconnections equal among different networks. The actual cost is determined by many technology-dependent issues and is therefore approximated via an appropriate model. Two measures of cost used in prior studies (for k-ary n-cubes or multidimensional meshes) are the bisection width and pin-out. We examine the bisection width and pin-out constraints first, and then we consider equalizing area based on the Thompson model [18, 19].

3.1 Constant Bisection Width Constraint

Dally [6] used bisection width as a measure of cost for the comparison of k-ary n-cubes. The bisection width of a network is the minimum number of wires cut when the network is divided into two equal halves. To equalize network costs, a larger channel width W is associated with networks that can be bisected by cutting fewer channels. The bisection widths for the mesh, hypercube, butterfly fat-tree and fat-pyramid (in units of channel width) are $\sqrt{n}$, $n/2$, $\sqrt{n}$ and $\sqrt{n}\log_{16} 4n$, respectively. (The least obvious result is the last, for which we add the number of channels to bisect the underlying fat-tree as well as each of the 2^h meshes at level h: $\sqrt{n} + \sum_{h=0}^{\log_2 \sqrt{n/4}-1} 2^h(\sqrt{n/4}/2^h)$.) The corresponding bisection widths (B) for a range of values of n are listed in Table 1. In each row, channel widths (W) are given to equalize the bisection width across different networks. For convenience, the channel width for the mesh is fixed at 32, and the other channel widths are rounded to the nearest integer to approximately equalize bisection width. (Channel width of 32 for the mesh is a reasonable value, since this would correspond to up to 128 connections per processor.)

3.2 Constant Pin-out Constraint

Abraham and Padmanabhan [1] used pin-out as a measure of cost. The pin-out of a node is the degree times the channel width W. For a k-ary n-cube, to maintain constant pin-out, a decrease in the dimensionality k mandates an increase in channel width W. The butterfly fat-tree and the fat-pyramid networks have different pin-outs for processor and switch nodes, but we can use the total pin-out, the sum of the pin-outs over all nodes. The total pin-outs for n-processor networks in units of channel width are $4(n-\sqrt{n})$, $n\log_2 n$, $(n+3(n-\sqrt{n}))$ and $(n+5(n-\sqrt{n})-\sqrt{n}\log_2 n)$ for the mesh, hypercube, butterfly fat-tree and fat-pyramid networks respectively. The total pin-out for the fat-tree is obtained by adding the pin-out of the processors to 6 times the sum of the number of switches at all the levels: $n + 6\sum_{h=0}^{\log_2 \sqrt{n/4}} 2^h(\frac{n}{4}/2^{2h})$. For the fat-pyramid, we must add an additional 4 connections for each switch but subtract off the missing connections at mesh edges: $n + 10\sum_{h=0}^{\log_2 \sqrt{n/4}} 2^h(\frac{n}{4}/2^{2h}) - 4\sum_{h=0}^{\log_2 \sqrt{n/4}} 2^h(\sqrt{n/4}/2^h)$. The number of pin-outs (PO) as described above are listed in Table 2 for a range of values of n. In each row, channel widths (W) are given to equalize the total pin-out across different networks.

3.3 Constant Layout Area Constraint

Anther constraint of interest is the layout area. With the layout area constraint, different networks are laid out on the same area with the same number of processors. The VLSI layout area is evaluated based on the assumption that all processors and switches are placed on a 2-D substrate. The substrate has two layer of interconnects for x-direction and y-direction respectively. Such a model can serve as a good abstraction for a variety of VLSI packaging technologies, such as wafer-scale integration or printed circuit boards [2]. More accurate modeling may require considering a complicated hierarchy of interconnection that consists of many levels, but the model here captures much of the complexity of interconnection, with more accuracy at least than looking only at bisection width.

n	*mesh*		*hypercube*		*BFT*		*fat-pyramid*	
	B	W	B	W	B	W	B	W
16	4	32	8	16	4	32	6	21
64	8	32	32	8	8	32	16	16
256	16	32	128	4	16	32	40	13
1024	32	32	512	2	32	32	96	11
4096	64	32	2048	1	64	32	224	9

Table 1: The bisection width with channel width 1 and the channel width to maintain constant bisection width across different networks.

n	*mesh*		*hypercube*		*BFT*		*fat-pyramid*	
	PO	W	PO	W	PO	W	PO	W
16	48	32	64	24	52	30	60	26
64	224	32	384	19	232	31	296	24
256	960	32	2048	15	976	31	1328	23
1024	3968	32	10240	12	4000	32	5664	22
4096	16128	32	49152	10	16192	32	23488	22

Table 2: The pin-out with channel width 1 and the channel width to maintain constant pin-out across different networks.

The layout area of a network includes the area for the processors and the area required due to interconnections. Since the area required for the processors is the same, we remove it from consideration and examine only the area necessary to achieve the interconnections. If processor area is very large relative to the wire pitch, it simply means we can scale up the channel widths of all the networks by a substantial factor.

For each of the networks, we can analyze the area by expressing the side length with n processors as

$$S(n) = \sqrt{n} \cdot d \cdot W \cdot P \tag{1}$$

where P is the wiring pitch (constant throughout our analyses) and d can be thought of as an average wire density per row or column (with channel width 1) when the processors are laid out in a $\sqrt{n}$ by $\sqrt{n}$ grid.

For the mesh $d = 1$. For the hypercube, we can use a layout as in [17], which is known to have optimal area. The maximum wire density per row is $d = \frac{2^{k+2}-(-1)^k-3}{6}$, where $k = \log_4 n$, which will be explained more fully in the final paper.

For the butterfly fat-tree and the fat-pyramid, we can obtain the side length from recurrence relations based on Figure 1. For the butterfly fat-tree, the recurrence is

$$S(n) = 2 \cdot S(n/4) + \sqrt{n} \cdot W \cdot P \tag{2}$$

with $S(4) = 2W \cdot P$. The solution is

$$S(n) = \sqrt{n} \log_4 n \cdot W \cdot P \tag{3}$$

so $d = \log_4 n$ for the butterfly fat-tree. Similarly, for the fat-pyramid $d = \frac{3}{2} \log_4 n$.

The wire densities as described above are listed in Table 3 for a range of values of n. In each row, channel widths (W) are given to equalize the area across different networks.

4 The Performance Comparisons

It has been customary to use network latency as the primary performance measure because of its tendency to limit performance in practice in today's fine-grained parallel systems. The maximum throughput is another important performance metric, and for certain applications such as sample sorting can be dominant [12]. The actual relationship of the execution time to the maximum throughput and average latency is determined by the nature of a parallel algorithm, and is beyond the scope of this paper. We will investigate both maximum throughput and average low load-rate latency for the networks under study.

The maximum throughput of a network is the maximum amount of messages that can come out of each

n	*mesh*		*hypercube*		*BFT*		*fat-pyramid*	
	d	W	d	W	d	W	d	W
16	1	32	2	16	2	16	3	10
64	1	32	5	6	3	10	4.5	7
256	1	32	10	3	4	8	6	5
1024	1	32	21	2	5	6	7.5	4
4096	1	32	42	1	6	5	9	4

Table 3: The wire density per row/column and the channel width under constant layout area constraints.

node in the network per clock cycle. Throughput varies with load-rate, which is the frequency with which processors generate messages. Throughput and load-rate are sometimes measured in fractions of the capacity per node of a network, where the capacity is defined as the maximum number of messages that can be in a network at one time. It is appropriate and convenient to use capacity as measures of throughput when comparing the performance of the same network under different operating conditions, or when comparing networks with the same capacities. However, since we are dealing with networks with different capacities, the actual number of bits per cycle per node is used as the basis for measuring load-rate and throughput.

As load-rate increases from zero, throughput follows load-rate until the load-rate has reached a certain value, when throughput starts to saturate. Additional increase in load-rate does not further increase throughput. As network size grows, the total throughput goes up, but the throughput per node generally decreases. For the mesh, butterfly fat-tree and fat-pyramid, this is mainly caused by the bisection width limitation, i.e., the bisection width doesn't scale up at the same ratio as does the number of processors. The hypercube network doesn't have bisection width limitation. However, when subject to the constraints (bisection width, pin-out, or area) to maintain equal cost, the hypercube network has to further reduce the width of each channel as network size grows. Moreover, the blocking of available channels by messages in the network also contributes to the decrease of the per node throughout.

The average latency, however, stays rather constant at low load-rates. As load-rate increases, average latency increase slowly until the network saturates, when the average latency increases rapidly due to the queuing time. In practice, parallel networks should be designed to operate on the the flat portion of the latency curve.

Figure 2 through Figure 4 show the simulation results for latency under constant bisection width, constant pin-out and constant area constraints respectively. Maximum throughput can be read from the latency graphs by looking for the load rate at which the network saturates. The results are obtained from network simulators for uniform random message patterns, using the channel widths calculated in Section 3. A worm length of 320 bits is used for all networks. Simulations are shown for several values of n up to $n = 4096$. Simulations for larger n are in progress.

Under the constant bisection width constraint, the mesh network always has the best maximum throughput. The hypercube network has the least throughput and the largest latency (in the graphs with 64 processors and up). This agrees with Dally's findings that the bisection width constraint strongly favors low-dimensionality [6]. The butterfly fat-tree network tends to achieve lower average latency than the mesh for large network size (≥ 256). The fat-pyramid network has lower throughout and higher latency compared with butterfly fat-tree because of its narrower channel width. It's performance is not far behind that of the fat-tree, but it remains poorer than the mesh at least for moderate network size.

When the constant pin-out constraint is used, the hypercube network shows average latency close to that of the mesh and fat-pyramid. The butterfly fat-tree again shows the lowest latency for $n \geq 1024$. The hypercube network has much higher throughput than other networks in agreement with Abraham and Padmanabhan's findings that constant pin-out favors high-dimensionality networks [1]. But the opposite conclusion is reached based on a comparison of low-load latency; here there is some disagreement with [1] due to the choice of a different routing model and other parameters. As the network size grows, there is also trend that the butterfly fat-tree and fat-pyramid show lower latency than the mesh.

With the constant area constraint, the butterfly fat-tree and the fat-pyramid networks show very close

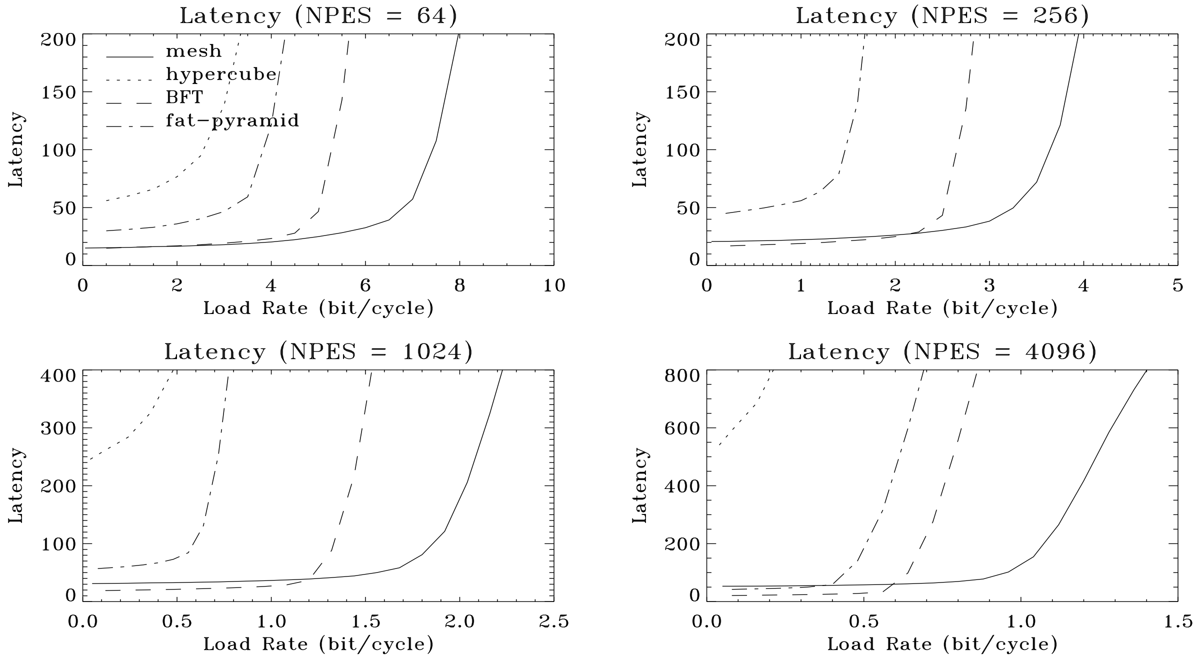

Figure 2: Throughput and latency comparison under constraint of equal bisection width.

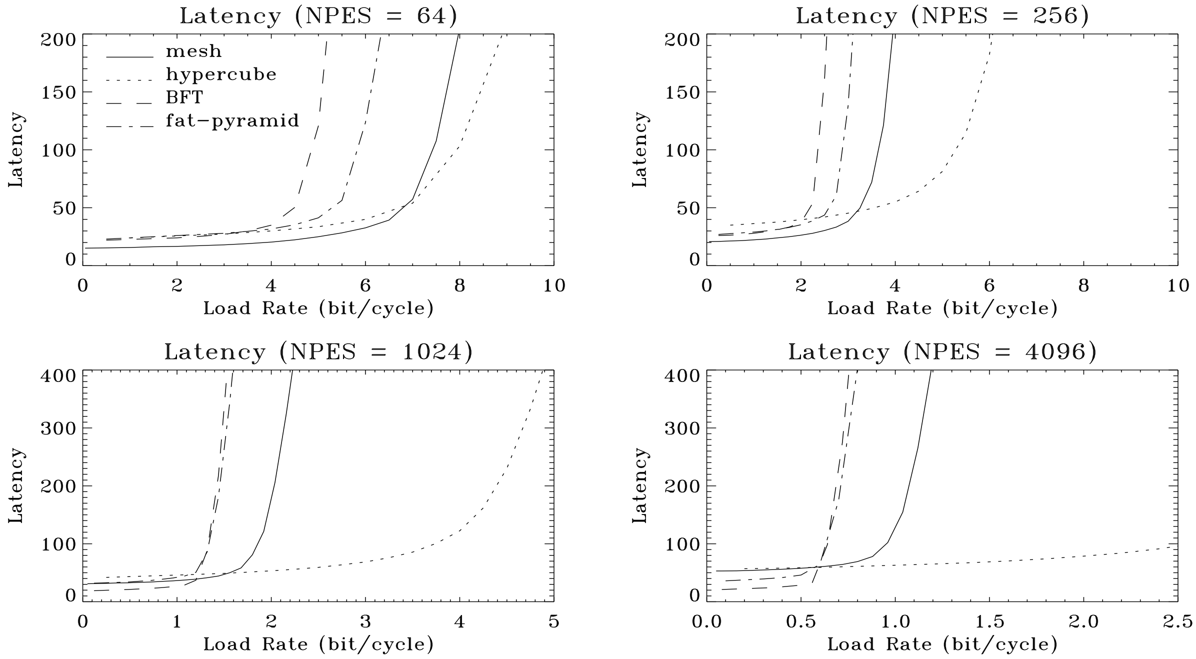

Figure 3: Throughput and latency comparison under constraint of equal pin-out.

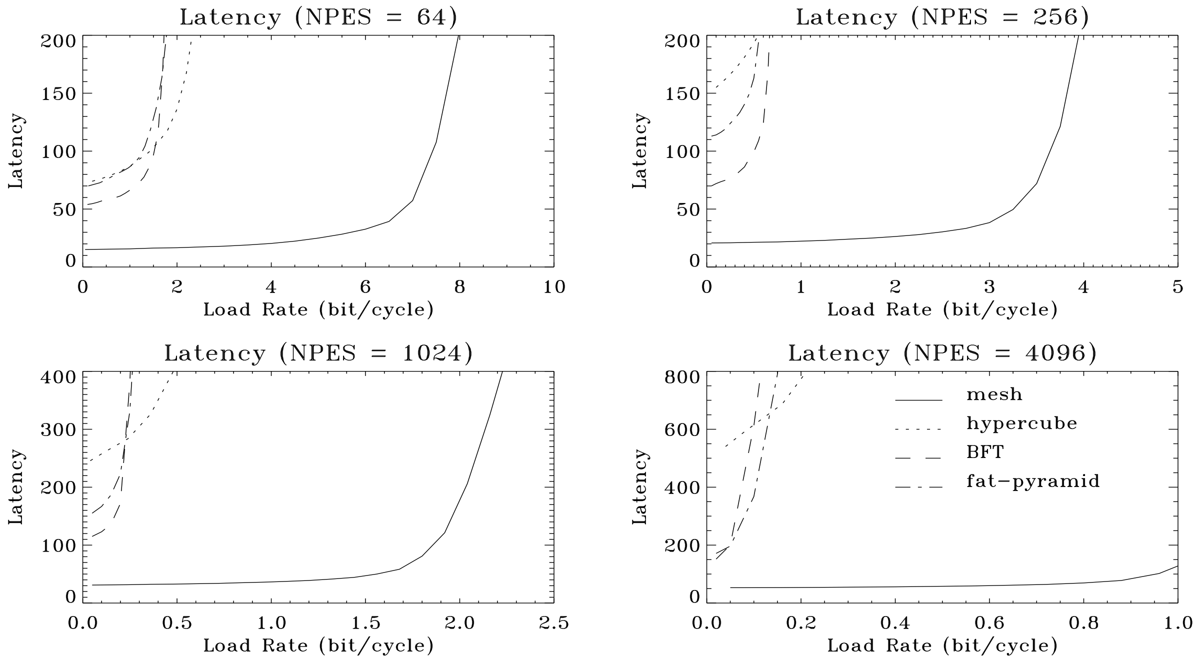

Figure 4: Throughput and latency comparison under constraint of equal interconnect area.

throughput, both small compared with those of the hypercube and mesh. The mesh network has the lowest average latency. The butterfly fat-tree and fat-pyramid have latencies between mesh and hypercube for network sizes of 256 and up. The poor showing of the fat-tree and fat-pyramid networks is probably substantially attributable to the use of the simple, basic structure illustrated in Figure 1, which use area a little larger than linear in the number of processors. We will perform further studies using the better variants of these networks that achieve linear area as discussed in [8, Section II].

5 Conclusions

The comparison of performance among different network architectures is critically affected by the constraint used to determine the available channel widths to equalize hardware cost. Layout area should be a more accurate measure than bisection width, and further study is called for using the more area-efficient versions of the fat-tree and fat-pyramid in [8] rather than just the simplified versions of the network considered so far. In addition, the area model is still a simplification of real packaging constraints. It would be desirable to more closely model this hierarchy of interconnection technologies.

Under constant bisection width and constant pin-out constraints the butterfly fat-tree achieves lower latency for $n \geq 1024$. The results obtained through simulation of random message pattern are more optimistic than the theoretical upper bounds under which some slowdown could occur [10, Section 3.1.1]. In addition, the butterfly fat-tree and fat-pyramid show close performance to each other under the unit wire delay model, but with a little poorer showing by the fat-pyramid. Since the purpose for introducing the fat-pyramid was to achieve favorable performance in comparison to other networks in the context of nonunit wire delay, additional empirical simulations should be performed with appropriate nonunit wire delay models.

The results presented in this paper (and many other papers) are based on uniform random message patterns. The performance of real parallel algorithms and the dependency of execution time on throughput and latency are topics of interest for future research.

References

[1] S. Abraham and K. Padmanabhan. Performance of multicomputer networks under pin-out constraints. *Journal of Parallel and Distributed Computing*, pages 237–248, Dec. 1991.

[2] H. B. Bakoglu. *Circuits, Interconnections, and Packaging for VLSI.* Addison-Wesley, 1990.

[3] P. Bay and G. Bilardi. An area-universal VLSI circuit. In *Proceedings of the 1993 Symposium on Integrated Systems*, pages 53–67, 1993.

[4] P. Bay and G. Bilardi. Deterministic on-line routing on area-universal networks. *Journal of the ACM*, 42(3):614–640, May 1995.

[5] J. Beecroft, M. Homewood, and M. McLaren. Meiko CS-2 interconnect Elan-Elite design. *Parallel Computing*, 20:1627–1638, Nov. 1994.

[6] W. J. Dally. Performance analysis of k-ary n-cube interconnection networks. *IEEE Trans. Computers*, 39(6):775–785, June 1990.

[7] W. J. Dally and C. L. Seitz. Deadlock-free message routing in multiprocessor interconnection networks. *IEEE Trans. Computers*, C-36(5):547–553, May 1987.

[8] R. I. Greenberg. The fat-pyramid and universal parallel computation independent of wire delay. *IEEE Trans. Computers*, 43(12):1358–1364, Dec. 1994.

[9] R. I. Greenberg and C. E. Leiserson. Randomized routing on fat-trees. In S. Micali, editor, *Randomness and Computation.* Volume 5 of *Advances in Computing Research*, pages 345–374. JAI Press, 1989.

[10] R. I. Greenberg and H.-C. Oh. Universal wormhole routing. In *Proceedings of the Fifth IEEE Symposium on Parallel and Distributed Processing*, pages 56–63. IEEE, 1993. Revised version submitted to *IEEE Transactions on Parallel and Distributed Systems.* Revised portions also appear as University of Maryland technical reports UMIACS-TR-93-60 and UMIACS-TR-93-102.

[11] R. I. Greenberg and H.-C. Oh. Universal wormhole routing. 1994. Submitted. Earlier versions of portions in University of Maryland technical reports UMIACS-TR-93-60 and UMIACS-TR-93-102 and *Proceedings of the Fifth IEEE Symposium on Parallel and Distributed Processing*, 1993.

[12] F. T. Hady. *A Performance Study of Wormhole Routed Networks Through Analytical Modeling and Experimentation.* PhD thesis, University of Maryland Electrical Engineering Department, 1993.

[13] M. W. Incorporated. Computing surface CS-2HA technical description, 1994.

[14] F. T. Leighton, B. M. Maggs, A. G. Ranade, and S. B. Rao. Randomized routing and sorting on fixed-connection networks. *Journal of Algorithms*, 17(1):157–205, July 1994.

[15] C. E. Leiserson. Fat-trees: Universal networks for hardware-efficient supercomputing. *IEEE Trans. Computers*, C-34(10):892–901, Oct. 1985.

[16] C. E. Leiserson, Z. S. Abuhamdeh, D. C. Douglas, C. R. Feynman, M. N. Ganmukhi, J. V. Hill, W. D. Hillis, B. C. Kuszmaul, M. A. S. Pierre, D. S. Wells, M. C. Wong, S.-W. Yang, and R. Zak. The network architecture of the connection machine CM-5. In *Proceedings of the 4th Annual ACM Symposium on Parallel Algorithms and Architectures*, pages 272–285. Association for Computing Machinery, 1992.

[17] A. G. Ranade and S. L. Johnsson. The communication efficiency of meshes, boolean cubes and cube connected cycles for wafer scale integration. In *Proceedings of the 1987 International Conference on Parallel Processing*, pages 479–482, 1987.

[18] C. D. Thompson. Area-time complexity for VLSI. In *Proceedings of the 11th ACM Symposium on Theory of Computing*, pages 81–88. ACM Press, 1979.

[19] C. D. Thompson. *A Complexity Theory for VLSI.* PhD thesis, Department of Computer Science, Carnegie-Mellon University, 1980.