

The Fat-Stack and Universal Routing in Interconnection Networks*

Kevin F. Chen and Edwin H.-M. Sha

Department of Computer Science

University of Texas at Dallas

Richardson, TX 75083, USA

kchen@student.utdallas.edu, edsha@utdallas.edu

Abstract

This paper shows that a novel network called the *fat-stack* is universally efficient when adequate capacity distribution is provided and is suitable for use as an interconnection network in parallel computers. The fat-stack resembles the fat-tree and the fat-pyramid in hardware structure, but it has its unique strengths. It is a construct of an atomic subnetwork unit consisting of one ring and one or more upward links to its upper subnetwork. This simple structure entails easy wirability. The network also uses less wires. More importantly, it has the capability to scale up to represent a large-scale distributed network. Our routing analysis shows that the fat-stack incurs certain unsurmountable delay when wires have uniform unit capacity assignment. But doubling capacity up the network forms a universal network. Our universality proof shows that a fat-stack variant with increased links and of area $\Theta(A)$ can simulate any competing network of area A with $O(\log A)$ overhead independent of wire delay. The universality result implies that the augmented fat-stack of a given size is nearly the best routing network of that size. The augmented fat-stack is the minimal universal network for an $O(\log A)$ overhead in terms of hardware usage. Actual simulations show that the performance of the augmented fat-stack approaches that of the fat-pyramid and is far higher than that of the fat-tree.

Key Words:

Fat-stack, interconnection networks, universality, VLSI

1 Introduction

An efficient network should move traffic speedily for the computing task and require no excessive hardware to build. In this paper we show that the *fat-stack* is such an efficient network by showing that it is *universal*, i.e. it can simulate any other network with an overhead of no more than (some power of) the logarithm of the area of the hardware containing the network. This universality result also implies that the fat-stack performs much better than or as well as most, if not all, of known networks. We will formally define the

fat-stack network in Section 2. We study the routing capability and universality of this tiered network.¹

Universal routing networks have been studied in the past in the context of interconnection networks. Work on the fat-tree and the fat-pyramid has initiated some principles and methodologies for studying and designing universal routing networks [5, 7, 13]. To apply these tenets to large-scale distributed networks, it is crucial for the network to be scalable. We propose the fat-stack because its structure is amenable to scaling. It turns out that the fat-stack is versatile as well. In this paper, we report the results of the fat-stack in the interconnection context. We wish this work also forms a good foundation for studying its application in the distributed setting. We will prove the universality of an augmented fat-stack using similar VLSI layout and wire delay conditions as those in [5, 7, 13, 16].

A typical fat-tree assumes a 4-ary tree structure with link capacities doubling up the levels of the tree. The fat-pyramid inherits the 4-ary tree framework of the fat-tree and adds a mesh on each level of the nodes up the tree. Specific hardware layouts of the two networks and a fat-stack will be described in Section 3. The fat-tree is the first proved universal network [13]. But it is universal only under unit wire delay condition; its universality does not hold under nonunit wire delays [5]. The fat-pyramid has been proven to be universal under both unit and nonunit wire delay conditions [5]. The fat-tree has been used in the CM-5 parallel computer whereas the fat-pyramid has not been adopted for any machine. Another clear advantage of the fat-pyramid over the fat-tree is its better absolute efficiency due to its hierarchical meshes. But these same meshes of the fat-pyramid reduce its scalability, increase its wire usage considerably, and make it not scalable to represent a distributed network.

The fat-stack is relatively simplistic in structure. It can be constructed by stacking up atomic subnetwork units following a fat-tree framework. The common subnetwork unit is made of a ring of certain nodes and one or more upward links each from one node of the unit. These links connect to the same node of a subnetwork right above the unit. The

¹The choice of the term “fat-stack” stems from the observation that the network is a construct of identical atomic subnetwork units stacked up and tapering upwards fast.

*Work supported in part by TI University Program, NSF EIA-0103709, Texas ARP 009741-0028-2001 and NSF CCR-0309461.

network is built up recursively. We consider two variants of the fat-stack in this paper. One has only one upward link from a subnetwork and the top level node is omitted. This variant is not strictly based on a tree due to the omission of some links. We refer to this variant as the *general fat-stack*. The second variant has as many upward links as the number of nodes in the subnetwork. We refer to this variant as the *augmented fat-stack* (AFS) which is the main focus of this paper.

The property of simple structure of the fat-stack entails easier wiring and packaging than the fat-pyramid and most other known networks. The network also uses less wires than the fat-pyramid. For a 4-ary layout, the fat-tree uses $2N - \sqrt{N}$ wires, the AFS uses $N/2 - \sqrt{N}$ more wires than the fat-tree, and the fat-pyramid uses $N/2 - \sqrt{N} \log_4 N$ more wires than the AFS, where N is the number of processors. In the layout, each leaf node is a single processor and the link capacities double up the levels.

In addition to scalability, less wire usage, and simpler construct, another advantage of the fat-stack over the fat-pyramid is that the AFS is the minimal universal network for the same asymptotic overhead of $O(\log A)$ under both unit and nonunit wire delay conditions, where A is the area of the hardware containing the network. This lower bound network notion is in terms of hardware usage. The AFS also has much better performance than the fat-tree due to its ring connections. The fat-stack can be adopted as a network structure when the computing tasks require no more than the capacity that the AFS can offer. It can also be considered as a design besides known universal network alternatives.

Routing schemes have direct impact on the universality of a network. We consider offline routing capability of the fat-stack. The universality of the fat-stack relies on routing capability in terms of a linear combination of congestion and distance that a packet travels. We say that network A can simulate network B with overhead μ if, for any t , the routing performed by B in time t can be performed by A in time μt .

While much of our work is analytical proof, we also simulated the AFS together with the fat-tree and the fat-pyramid in order to visualize and compare their performances in realistic terms. The simulations were carried out using the $ns-2$ network simulator. The nodes of the networks are thought as fitted on the same template of reference, and the wires have nonunit delay. This type of simplification enables us to relate the experimental data to the analytical results. The simulations demonstrate that the AFS has a high performance improvement over the fat-tree and approaches the fat-pyramid in efficiency. The universal routing capability of the AFS makes it a good alternative interconnection network.

Our contributions are thus the following: (1) We added another universal network class into the repertory of known universal networks. (2) The AFS is the minimal universal network under nonunit wire delay condition with $O(\log A)$

overhead. (3) The fat-stack is easier to construct and easier to scale than the fat-pyramid. (4) The general fat-stack can scale to the size of a distributed network, so analysis results would be valuable to evaluate the performance of large-scale distributed networks.

2 Network Model

We now describe the graphic-theoretic topology of the fat-stack and a computation and communication framework on which routing is based.

The fat-stack is a hierarchical network, consisting of tiers or levels. Each level has one or more subnetworks. Each subnetwork is a ring. There is one link connecting a subnetwork to an upper level from one node to an upper node. A fat-stack can have arbitrary levels of rings. In our analysis of the fat-stack application as interconnection network, we use the AFS variant in which every node of a subnetwork has a link to a common upper level node. A graphical representation of an AFS is shown in Figure 1.

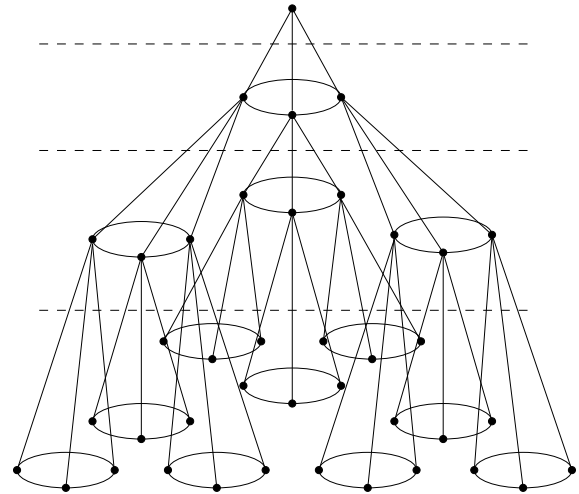


Figure 1: Topology of an AFS. Each subnetwork has three nodes. Dashed lines represent tier boundaries.

Let n be the number of nodes in a subnetwork. The base structure of an AFS is one of an n -ary tree in which each non-leaf node represents a switch and a leaf node represents a processor. An AFS is formed by adding a ring connecting the child nodes of each parent in the tree.

By convention, an edge in the network graph corresponds to a channel. A channel is composed of a certain number of wires. Each wire corresponds to one atomic unit of capacity. In hardware layout graphs, capacity is represented differently. Each edge represents a wire of unit capacity. To increase capacity, extra edges and nodes of the same capacity are created. We will discuss one hardware layout in the next section.

The capacities of the channels and the nodes of a fat-tree are defined as doubling from one level to the next up the

tree and growing a little less than doubling at levels close to the root [13]. The normal form of the fat-pyramid doubles the capacities up the tree [5]. The butterfly fat-tree which is what the fat-pyramid is based on has also a prefixed capacity distribution [7]. We will use doubling capacities for the AFS.

As demonstrated in [14], an arbitrary number of capacity distributions can be specified for the fat-stack by combining two types of switches with a constant number of inputs and outputs.

We employ an abstract model of packet routing and operation. The basic mode of operation assumed of the fat-stack is the usual distributed random-access machine (DRAM) model [15]. In a DRAM, all memory is located at the processors and its access is made by messages routed through the routing interconnection network. Indivisible packets will be used for routing analysis and they can be perceived as the basic constituents of data traffic and as a convenient scalar quantity for mathematical analysis. Large messages can be fragmented into packets.

3 Universality of the Augmented Fat-Stack

In this section, we prove the universality of the AFS with increased link capacities. The added capacities are necessary for the network to be universal. The only limitation to the universality is that the competing network has to have the same total area as the AFS. The AFS can be compared to the fat-tree and fat-pyramid networks and hence enable us to prove its universality following a similar approach. Its universality makes the AFS a good candidate as the interconnection network for a general-purpose parallel computer.

Our proofs will be based on the general but powerful routing results obtained by Leighton, Maggs, and Rao [9–12]. The results pertain to offline algorithms that work under unit wire delay condition. Unit wire delay denotes that it takes unit time for a packet to move through a wire. This assumption implies that a packet traverses a distance of at most one during a single routing step or one unit time, and that at most one packet can pass through a wire during one routing step. In later analysis, wires will be considered as *transmission lines* that pipeline bits (packets) and have delay (speed) variations. The following lemma will suffice to prove the universality of the fat-stack. In the lemma, the term *congestion* refers to the maximum number of packets that must traverse a single edge in one direction, and *dilation* refers to the maximum number of edges that must be traversed by a single packet.

Lemma 3.1 (Leighton et al.). *For any set of packets with edge-simple paths having congestion c and dilation d , there is a schedule of length $O(c+d)$ requiring a maximum queue size of $O(1)$.*

To compare with the fat-tree and the fat-pyramid, the AFS will assume a similar hardware configuration as follows: (1)

it is based upon a 4-ary tree in singular representation; (2) the capacities of nodes and links are modulated as doubling up the tiers; (3) we assume that the leaf nodes are not connected with each other and each leaf node is an H-tree layout of $\log_2 A$ processors each of area $\Theta(\log A)$; and (4) a procedure of “split-and-duplicate” of nodes and links is performed on the structure shown in Figure 1 to obtain an adequate interconnection layout.

It has been shown that a fat-tree can *efficiently* (i.e. in no more than polylogarithmic slowdown) simulate any network of comparable volume or area under the unit wire delay assumption [1,5,7,9,13], and that the fat-pyramid can achieve a logarithmic efficiency under a nonunit (linear) wire delay model [5]. There has been some variation in the exact number of links and the overall connectivity implemented in the models of the fat-tree since its universality was first proved by Leiserson [13]. We refer mainly to the results proved by Greenberg along with those for the fat-pyramid [5]. The fat-pyramid is based on superimposing hierarchical mesh connections on a butterfly fat-tree.

We first establish the similarity between a typical butterfly fat-tree and an AFS and prove the universality of the AFS under unit wire delay assumption. We then prove that the AFS is universal under nonunit wire delay condition due to its added ring connectivity. To facilitate our discussion, we illustrate the fat-tree, the fat-pyramid, and the AFS together in Figure 2, which is adapted from Figure 1 of [5].

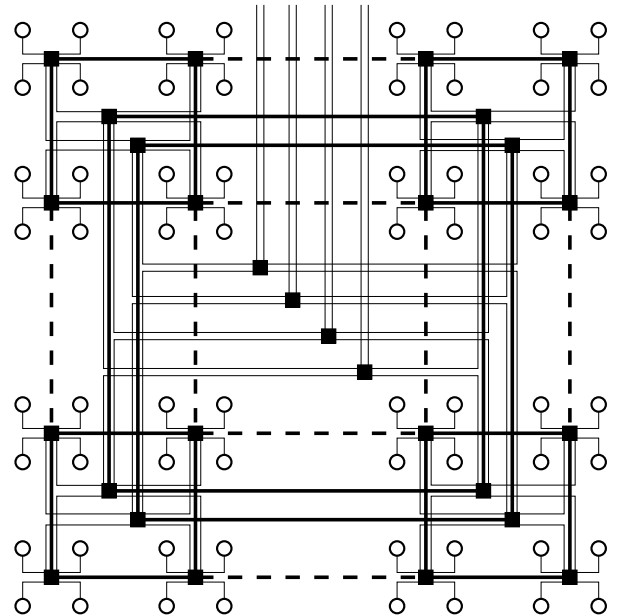


Figure 2: A fat-tree, a fat-pyramid, and an AFS in one layout. Processors are represented by circles; the squares are switches. The fat-tree links are represented by thin lines, the fat-pyramid mesh links by thick solid and dashed lines, and the AFS ring links by the thick solid lines.

The channel capacities in the fat-tree in Figure 2 double

at increasing levels of the underlying 4-ary tree inasmuch as all 4 child nodes connect to one parent node with a double redundancy at each level. This capacity increment can be viewed as splitting the upper node into two and duplicating the four downward links once.

We can do a reverse of the “split-and-duplicate” procedure to transform the fat-tree into a singular 4-ary tree structure by overlaying individual nodes and links and hence their capacities. We can perform the same overlay procedure on the fat-pyramid to get a 4-ary tree in which all the switch nodes at each level of the underlying tree are connected within a mesh. The procedure can also collapse the AFS layout into a singular structure similar to the one shown in Figure 1.

The difference of the three networks lies in if and how the nodes at a level of the tree are interconnected. Unlike the fat-pyramid, the fat-tree has no connecting links among switches on the same levels, and the AFS features the switch nodes of the same parent connected by a ring. The degree of connectivity decreases in the order of the fat-pyramid, the AFS, and the fat-tree.

In overlaying the nodes and links of the fat-pyramid, the capacities of the mesh links increase at the same rate as those of the other nodes and links. The overall capacity increase is 2^h times up the underlying fat-tree where h is the level number from the leaf nodes designated as level 0. We modulate the AFS with the same capacity distribution pattern as for the fat-tree and the fat-pyramid. Intuitively, compared to the fat-tree, the AFS merely possesses more connectivity due to the connected subnetwork nodes and thus can not be worse off in computing and routing efficiency.

Universality proofs for the fat-tree and fat-pyramid are parameterized by hardware area or volume. A geometric bisecting method is used to decompose the competing network to match with the structure of the base network [5, 13]. We shall adopt this method in our proofs. This method has its basis in the theories and constructions of VLSI graph layouts [2, 3, 8]. The competing network can be in terms of a cube or an area in a two dimensional design space. Extension of computation analysis from two dimensions to three dimensions has been shown to be straightforward [4, 6].

The decomposition is to recursively cut the cube or area into two pieces in the direction of the shorter edges until each piece contains either zero or one processors. It has been proven that a balanced decomposition tree can always be obtained, in which the number of processors on either side of a given node (cut) is equal to within one [13].

In view of the decomposition tree, a valid assumption is that the number of packets that can enter or leave an area (equivalently a subtree) in unit time is proportional to the perimeter of the area. As such, there are $O(\log A)$ packets amortized on each wire of the fat-tree of area $\Theta(A)$, resulting in an overhead. With this postulate and based on the general routing theorem of Lemma 3.1, Greenberg [5] proved the universality of the fat-tree in terms of hardware

area.

Lemma 3.2 (Greenberg). *A fat-tree of area $\Theta(A)$ can simulate any network of area A with $O(\log A)$ overhead.*

Adding a ring on each subnetwork of the fat-tree can cause only a constant factor increase in area. In addition, the connections and the framework of the fat-tree are intact when the rings are added. Thus, we have the following theorem.

Theorem 3.1. *An AFS of area $\Theta(A)$ can simulate any network of area A with $O(\log A)$ overhead.*

The benefit of the AFS rings is that they enable us to drop the unit wire delay assumption used in arriving at the above theorem. The fat-tree is not universal when simulating a mesh under nonunit wire delay assumption [5].

We now proceed to prove the universality of the AFS under nonunit wire delay condition. We again base our proof on the result of Greenberg pursuant to the fat-pyramid. We shall again adopt the condition that each bottom level node is a cluster of $\Theta(\log A)$ processors in an H-tree layout.

In the fat-pyramid proof, an appropriate layout of the fat-pyramid is produced by embedding the base butterfly fat-tree into the graph of the tree of meshes, performing a “fold-and-squash” transformation, and adding the mesh connections of the fat-pyramid [5]. In this layout, the length of the edges of the tree of meshes becomes $O(\log A)$ whereas the length of the wires connected to a switch h levels up from the H-tree blocks are $O(2^h \log A)$. Now the routing path of a packet goes along fat-tree edges upwards until it reaches a switch that is adjacent in the same mesh to another switch that leads to the destination by going down fat-tree edges. Two processors separated by a distance δ in the competing network R is $\lceil \delta / \log_2 A \rceil$ H-tree blocks apart in the fat-pyramid F . A packet must traverse the height of the subtree enclosing the two processors ($\log_2(\delta / \log A)$ levels) and the H-tree blocks. It travels a distance of $O(\delta + \log A)$. It is thereof proved that any message following a path of length δ in a competing network travels $O(\delta + \log A)$ distance in the fat-pyramid.

The congestion in F to route the same message set as in R is $O(T \log A)$ where T is the time required to deliver the message set in R and, given a linear delay function w , $T \geq w(\delta)$. A message must traverse at most $2 \log_2 \delta$ fat-pyramid edges and each edge has a maximum delay of $w(\delta + \log_2 A)$ steps. Hence the following theorem is proved by computing the overhead based on the total delivery times in F and R and the definition of w [5].

Lemma 3.3 (Greenberg). *Using transmission lines, a fat-pyramid of area $\Theta(A)$ can simulate any network of area A with $O(\log A)$ overhead under nonunit wire delay assumption.*

We can now show the universality of the AFS by following the same lines of the fat-pyramid proof. In the AFS layout a packet has to travel a path of the same length as for the

fat-pyramid to reach a switch that is adjacent (at most two edges apart) to another switch in the same mesh (ring) that leads to the destination node, and hence the same overall distance as in the fat-pyramid layout. Similarly, the congestion and the dilation are the same. Note that if the layout is n -ary where $n > 4$, the packet can get to the destination via the root of the underlying n -ary tree covering the source and destination processors by traveling two edges, the same as taking two mesh edges in a 4-ary layout. The below theorem follows.

Theorem 3.2. *Using transmission lines, an AFS of area $\Theta(A)$ can simulate any network of area A with $O(\log A)$ overhead under nonunit wire delay assumption.*

Note that the path that a packet in the AFS travels could not be shorter for the given connectivity, so the AFS is the minimal universal network with $O(\log A)$ overhead in terms of wire usage.

4 Experimental Results

We did several experiments using the *ns-2* network simulator to visualize and compare the performance of the networks discussed so far. There are two fixtures in the experiments. The nodes are considered as fixed on a template, and wiring the various links for each network does not bloat the template. This makes it easy to stipulate that the networks are of the same area. A fixed traffic scenario of three traffic types is used to run traffic through each of the networks in order to obtain data on how well each performs.

We simulated the fat-tree, the augmented fat-stack, and the fat-pyramid. The general template is similar to the node layout illustrated in Figure 2. Now the switches of the second and top levels are combined into singular nodes. Wires are also combined into singular links with composite capacities. Each network has a distinct wire layout. In the template there are 64 processor nodes and 21 switches as required by a 4-node subnetwork configuration. Downward links from the second level switches have a capacity of 0.2 Mbps. Links from the top switch are 0.4 Mbps in capacity. Upward links from the bottom level have a capacity of 0.1 Mbps. Links within subnetworks on a level have the same capacities as their upward links.

The split wires and nodes used previously in discussing the networks are collapsed into singular links and nodes with composite capacities. Link connections differ among the networks. However, their topologies can all be easily fitted into the template without violating the area constraint. Leaf nodes consist of single processors that act as switches as well as processors, and they are connected within a mesh in the fat-stack and the fat-pyramid. The links are modeled as duplex links having a delay of 10 ms. This implies that the wires have varying (nonunit) delays and varying length as the materials of the wires could vary. From the processors to the switch at the top in the first four networks, there

are three tiers. In each tier, links in the subnetworks and its upward link(s) are assigned the same capacity. Upwards across the three tiers, link capacities are 0.1, 0.2, and 0.4 Mbps successively, i.e., they double up the underlying 4-ary tree.

A simplistic communication model was used in all of the simulations. We used UDP as the transport protocol, so in effect the simulations merely emulate IP routing. We used the adaptive link-state routing as the routing protocol so that routing responds to traffic loads on the links. Each network was simulated under all three traffic types. The duration of the simulations is all 120 seconds. During this period, 40 processor nodes generate traffic at start and end times generated at random. Traffic sinks are also randomly selected among the 64 processor nodes. There are a total of 60 connections or flows created during each simulation. The flows start and end at random times within the simulation duration.

The types of the traffic generated at the sources are of constant bit rate (CBR), exponential on/off, and Pareto on/off. The common attributes of the traffic types are that the packet size is 1,000 bytes and the average sending rate is 80 Kbps. Other attributes vary with the latter two types having bursty sending patterns. The queuing rule applied for all links is Stochastic Fair Queuing with default configuration parameters (maximum queue limit set at 40 packets and using 16 buckets for hashing each of the flows).

We extracted from the results of experiments the overall throughput and delay as the specific traffic moves through each network. They are calculated by obtaining the start time and the duration that each packet stayed in the network before arriving at its final destination. We use them as the metrics for comparison. The data are listed in Table 1.

Table 1: Simulation Results

Network	Traffic Type	Throughput (bps)	Delay (sec)
Fat-tree	CBR	928113.333333	0.763641
	Exponential	917320.000000	0.841578
	Pareto	875986.666667	0.780242
Aug. fat-stack	CBR	1126821.333333	0.194550
	Exponential	1130601.333333	0.230433
	Pareto	1097736.000000	0.215983
Fat-pyramid	CBR	1652286.666667	0.146700
	Exponential	1624204.000000	0.169914
	Pareto	1589376.000000	0.162286

The asymptotic nature of the simulation overheads we derived in Section 3 for networks does not warrant a direct comparison of these numbers in terms of $O(\log A)$ although it is inferable that $\log A$ corresponds to $\log N$ (A is greater than N). Therefore, without obscuring our analytical results, we can rate the performance of the networks relative to each other.

The data show that throughput increases and delay decreases in the order of the fat-tree, the augmented fat-stack,

and the fat-pyramid. This is because more and more mesh links are added in that order. The augmented fat-stack performs better than the fat-tree by $\sim 23\%$ in throughput and $\sim 73\%$ in overall delay. It is less efficient than the fat-pyramid by $\sim 31\%$ in throughput and $\sim 25\%$ in overall delay. In other words, the augmented fat-stack has a considerable improvement over the fat-tree and approaches the fat-pyramid in efficiency. Note that the fat-pyramid uses much more wires and thereby incurs more complexity in wiring and packaging. In addition, the fat-pyramid does not scale to represent a distributed network.

5 Conclusion

In this paper we have shown that the augmented fat-stack with upward increasing link capacities is universally efficient and is suitable for use as an interconnection network. The universality is based on offline routing capability. But online routing should have the same efficiency due to analysis of packet routing on a “leveled” network [9, 11].

Each of the two fat-stack variants can be viewed as a construct of a common atomic unit consisting of a ring and one or more upward links. The augmented fat-stack has simpler connectivity than the fat-pyramid and is therefore easier for wire routing and uses less hardware. In simulations, the augmented fat-stack performs almost as well as the fat-pyramid and much better than the fat-tree. This makes it a valuable candidate for certain classes of supercomputing tasks that would require an interconnection network more efficient than the fat-tree. In some cases, the augmented fat-stack can be a substitute for the fat-pyramid.

The general fat-stack variant incurs some unsurmountable delay as each of its subnetworks has only one upward link. However, the general fat-stack can be scaled up to represent a distributed network. It would be desirable to study the scalability of the fat-stack and its application to distributed networking.

References

- [1] P. Bay and G. Bilardi. Deterministic on-line routing on area-universal networks. *Journal of the ACM*, 42(3):614–640, May 1995.
- [2] S. N. Bhatt and F. T. Leighton. A framework for solving VLSI graph layout problems. *Journal of Computer System Sciences*, 28:300–343, Apr. 1984.
- [3] S. N. Bhatt and C. E. Leiserson. How to assemble tree machines. In F. P. Preparata, editor, *VLSI Theory*, volume 2 of *Advances in Computing Research*, pages 95–114. JAI Press, 1984.
- [4] R. I. Greenberg. *Efficient Interconnection Schemes for VLSI and Parallel Computation*. PhD thesis, Massachusetts Institute of Technology, Aug. 1989. MIT/LCS/TR-456.
- [5] R. I. Greenberg. The fat-pyramid and universal parallel computation independent of wire delay. *IEEE Transactions on Computers*, 43(12):1358–1364, Dec. 1994.
- [6] R. I. Greenberg and C. E. Leiserson. A compact layout for the three-dimensional tree of meshes. *Applied Mathematics Letters*, 1(2):171–176, 1988. Also see erratum in vol. 1, no. 3, p. 315.
- [7] 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.
- [8] F. T. Leighton. A layout strategy for VLSI which is provably good. In *Proceedings of the 14th Annual ACM Symposium on Theory of Computing*, pages 85–98, May 1982.
- [9] 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, 1994.
- [10] F. T. Leighton, B. M. Maggs, and S. B. Rao. Packet routing and job-shop scheduling in $O(\text{congestion} + \text{dilation})$ steps. *Combinatorica*, 14(2):167–186, 1994.
- [11] T. Leighton, B. Maggs, and S. Rao. Universal packet routing algorithms. In *Proceedings of the 29th Annual Symposium on Foundations of Computer Science*, pages 256–269. IEEE Computer Society Press, 1988.
- [12] T. Leighton, B. Maggs, and A. W. Richa. Fast algorithms for finding $O(\text{congestion} + \text{dilation})$ packet routing schedules. *Combinatorica*, 19(3):375–401, 1999.
- [13] C. E. Leiserson. Fat-trees: universal networks for hardware-efficient supercomputing. *IEEE Transactions on Computers*, C-34(10):892–901, Oct. 1985.
- [14] C. E. Leiserson. VLSI theory and parallel supercomputing. In C. L. Seitz, editor, *Advanced Research in VLSI: Proceedings of the Decennial Caltech Conference on VLSI*, pages 5–16. MIT Press, 1989.
- [15] C. E. Leiserson and B. M. Maggs. Communication-efficient parallel algorithms for distributed random-access machines. *Algorithmica*, 3:53–77, 1988.
- [16] V. Strumpfen and A. Krishnamurthy. A collision model for randomized routing in fat-tree networks. Technical Memo MIT-LCS-TM-629, Laboratory for Computer Science, Massachusetts Institute of Technology, 15 July 2002.