# Low Latency, High Bisection-Bandwidth Networks for Exascale Memory Systems

Shang Li, Po-Chun Huang, David Banks, Max DePalma, Ahmed Elshaarany,
Scott Hemmert*, Arun Rodrigues*, Emily Ruppel, Yitian Wang,
Jim Ang*, and Bruce Jacob

Electrical & Computer Engineering
University of Maryland
College Park, Maryland, USA
{shangli,hpcalex,blj}@umd.edu

* Scalable Computer Architectures
Sandia National Laboratories
Albuquerque, New Mexico, USA
{kshemme,afrodri,jaang}@sandia.gov

## ABSTRACT

Data movement is the limiting factor in modern supercomputing systems, as system performance drops by several orders of magnitude whenever applications need to move data. Therefore, focusing on low latency (e.g., low diameter) networks that also have high bisection bandwidth is critical. We present a cost/performance analysis of a wide range of high-radix interconnect topologies, in terms of bisection widths, average hop counts, and the port costs required to achieve those metrics. We study variants of traditional topologies as well as one novel topology. We identify several designs that have reasonable port costs and can scale to hundreds of thousands, perhaps millions, of nodes with maximum latencies as low as two network hops and high bisection bandwidths.

## CCS Concepts

•**Networks** → **Network architectures;** •**Hardware** → **Buses and high-speed links;**

## Keywords

network topology; supercomputer design; SST

## 1. INTRODUCTION

Computational efficiency is the fundamental barrier to exascale computing, and it is dominated by the cost of moving data from one point to another, not by the cost of executing floating-point operations [11, 16]. Data movement has been the identified problem for many years (e.g., "the memory wall" is a well-known limiting factor [21]) and still dominates the performance of real applications in supercomputer environments today [13]. In a recent talk, Jack Dongarra showed the extent of the problem: his slide, reproduced in Figure 1, shows the vast difference, observed in actual systems (the top 20 of the Top 500 List), between peak FLOPS,

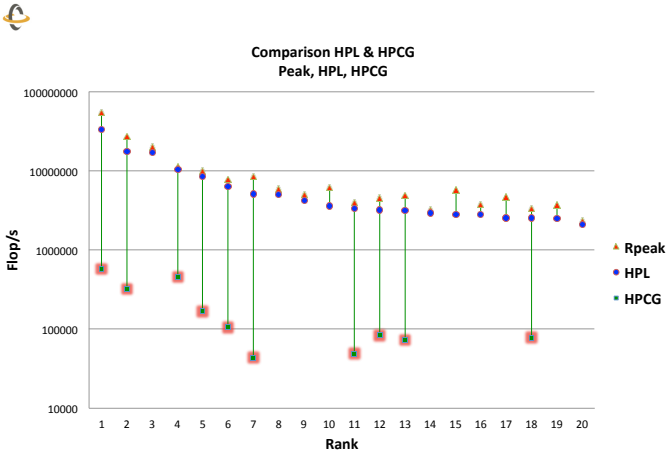


**Figure 1: A comparison of max theoretical performance, and real scores on Linpack (HPL) and Conjugate Gradients (HPCG). Source: Jack Dongarra**

the achieved FLOPS on Linpack (HPL), and the achieved FLOPS on Conjugate Gradients (HPCG), which has an all-to-all communication pattern within it. While systems routinely achieve 90% of peak performance on Linpack, they rarely achieve more than a few percent of peak performance on HPCG: as soon as data needs to be moved, system performance suffers by orders of magnitude.

Thus, to ensure efficient system design at exascale-class system sizes, it is critical that the system interconnect provide good all-to-all communication: this means high bisection bandwidth and short inter-node latencies. Exascale-class machines are expected to have on the order of one million nodes, with high degrees of integration including hundreds of cores per chip, tightly coupled GPUs (on-chip or on-package), and integrated networking. Integrating components both increases inter-component bandwidth and reduces power and latency; moreover, integrating the router with the CPU (concentration factor $c = 1$) reduces end-to-end latency by two high-energy chip/package crossings. In addition to considering bisection and latency characteristics, the network design should consider costs in terms of router ports, as these have a dollar cost and also dictate power and



*MEMSYS '16, October 03 - 06, 2016, Alexandria, VA, USA*



ISBN 978-1-4503-4305-3/16/10…$15.00

DOI: http://dx.doi.org/10.1145/2989081.2989130

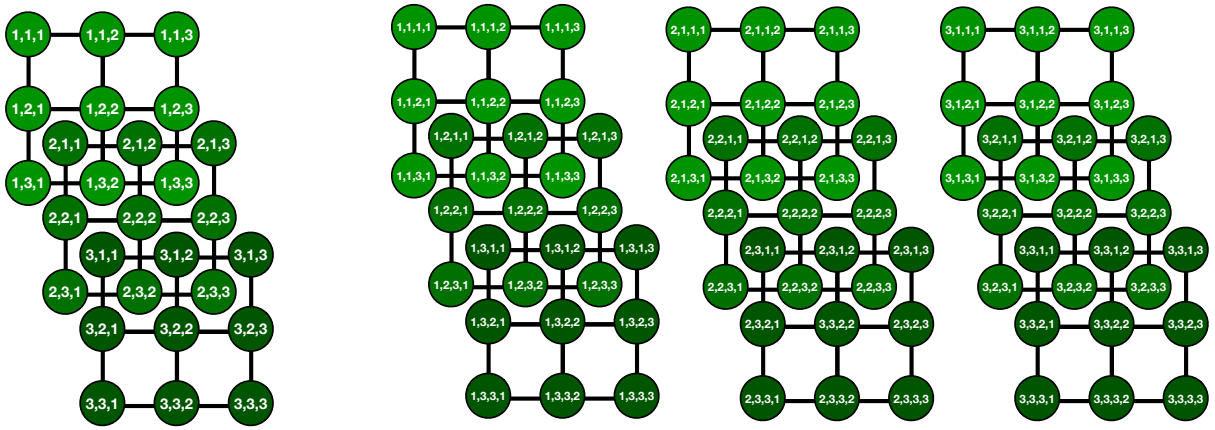


**Figure 2: 2D, 3D, and 4D meshes/tori. Left: a 3D mesh/torus of side n=3 is simply 3 copies of the 2D mesh/torus tiled into a new dimension. Right: continuing this process produces meshes/tori in 4D, 5D, 6D, and higher dimensions. Note: numerous links not shown, for visibility.**

energy overheads.

We present a cost/performance analysis of several high-radix network topologies, evaluating each in terms of port costs, bisection bandwidths, and average latencies. System sizes presented here range from 100 nodes to one million. We find the following:

- Perhaps not surprisingly, the best topology changes with the system size. Router ports can be spent to increase bisection bandwidth, reduce latency (network/graph diameter), and increase total system size: any two can be improved at the expense of the third.
- Flattened Butterfly networks match and exceed the bisection bandwidth curves set by Moore bounds and scale well to large sizes by increasing dimension and thus diameter.
- Dragonfly networks in which the number of inter-group links is scaled have extremely high bisection bandwidth and match that of the Moore bound when extrapolated to their diameter-2 limit.
- High-dimensional tori scale to very large system sizes, as their port costs are constant, and their average latencies are reasonably low (5 - 10 network hops) and scale well.
- Novel topologies based on Fishnet (a method of interconnecting diameter-2 subnets) become efficient at very large sizes - hundreds of thousands of nodes and beyond.

Our findings show that highly efficient network topologies exist for tomorrow's exascale systems. For modest port costs, one can scale to extreme node counts, maintain high bisection bandwidths, and still retain low network diameters.

## 1.1 Network Topologies Considered

The following describes the various topologies evaluated in this paper. Due to the desire for good all-to-all communication patterns, we focus on high-radix networks, as opposed to low radix topologies such as fat trees [12].

### *Hypercube, Torus, and Higher-Dimensional Extensions*

2D and 3D meshes and tori are well known and are relatively common. We scale the dimensionality of the tori from 2D to 10D (see Figure 2 ) and note that any torus with two nodes per side is a $de facto$ hypercube. Like hypercubes, tori have very good efficiency at the higher dimensions: they have fixed port costs (each router has a fixed number of ports, no matter how large the network), and their average latency values grow more slowly than the pin costs of most high-radix network topologies. In addition, they have simple routing heuristics: a node's address directly determines which port a router can use, without need for a table lookup.

In general, the graphs have the following characteristics, where $D$ is the dimension, and $n$ is the length of a side (for the sake of simplicity, we assume all sides are of equal length):

- Nodes: $n^D$
- Ports: $2D$ (except for $n = 2$, which is $D$)
- Bisection Links: $2n^{D-1}$ (except for $n = 2$, which is $n^{D-1}$)
- Maximum Latency: $\sim Dn$

The maximum latency depends on actual configuration, such as whether the length of a side $n$ is an even number or odd.

Note that, in the degenerate case of $n = 2$ (a hypercube), a wraparound link is not needed, as there are only two nodes to a given side; for this case, the ports per node and the bisection bandwidth are each reduced by half compared to those for a torus.

### *Flattened Butterfly and Higher-Dimensional Extensions*

The Flattened Butterfly interconnect shown in Figure 3 provides a regular structure for a two-hop network, which is beneficial because it simplifies routing heuristics: like tori, a node's address determines which port a router uses, obviating a table lookup. A Flattened Butterfly network is created by combining nodes in both horizontal and vertical dimensions into fully connected graphs. Thus, any node in

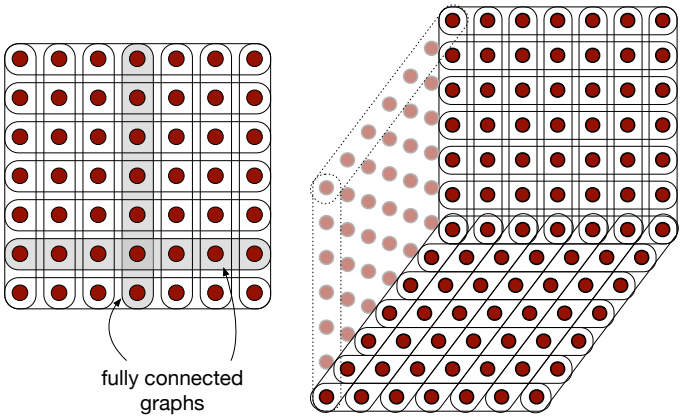


**Figure 3: Flattened Butterfly (left) and 3D Flattened Butterfly (right): each dimension is characterized by fully connected graphs.**

a Flattened Butterfly topology lies at a maximum distance of 2 from any other node in the network. Noting that this is a 2D structure, it is straightforward to extend the concept into three dimensions and beyond. For example, Figure 3 shows both the traditional 2D topology and a 3D Flattened Butterfly structure as well.

In general, the graphs have the following characteristics, where $D$ is the dimension, and $n$ is the length of a side (as with the tori, we assume all sides are of equal length):

- Nodes: $n^D$
- Ports: $D(n1)$
- Bisection Links: $\sim n^{D-1}(n/2)^2$
- Maximum Latency: $D$

The number of bisection links depends on configuration, such as whether the length of a side $n$ is an even number or odd.

### *Moore Graphs*

Computer networks based upon the Moore limit have been used since the 1960s [7, 3] and have the advantage of maximizing the number of nodes in the system, given a specific number of ports per router and a desired maximum network hop count (graph diameter). Compared to regular topologies such as meshes/tori and Flattened Butterfly networks, their packet-routing heuristics are more complex and take more overhead, because a table lookup is required. However, a Moore graph requires fewer ports than a Flattened Butterfly to connect the same number of nodes in the same number of hops.

Figure 4 illustrates two extremely well-known instances of two-hop (diameter 2) Moore graphs, in which every node lies at a distance of at most two hops from every other node in the graph. The Petersen graph combines 10 nodes, each of which has 3 ports. The Hoffman-Singleton graph combines 50 nodes, each of which has 7 ports, and, as the figure shows, the Hoffman-Singleton graph contains within it five disjoint copies of the Petersen graph. This is important, as it has manufacturing implications: only one circuit board design would be needed for the subnetworks.

Moore graphs were not generally used in network design for decades, designers preferring hardware routers that use a node's address to drive routing heuristics rather than a table lookup. However, Moore graphs have seen a recent resurgence in popularity. In 2006, the Petersen graph was used to construct a high-performance DSP cluster, an image processing system for a NASA satellite [17]. In 2013, Bao [2] proposed using 2-hop Moore graphs as interconnect networks. In 2014, Besta [4] proposed using 2-hop Moore graphs as interconnect networks, calling the configuration "Slim Fly."

The benefit of using a diameter-2 Moore graph is the maximum system-wide latency of two hops. This comes at a port cost, as to reach higher node counts, the number of ports grows rapidly. For instance, 16 ports per router are required for a network of 198 nodes, and 79 ports are needed to build a network of 5600 nodes.

Moreover, the irregular nature of Moore graphs makes their construction at large sizes non-trivial. The first two examples provided in Figure 4 are the best known precisely because they are the only two regular variants ever discovered: these are the only two graphs known to achieve the Moore limit (the number of nodes reachable given a maximum hop count and number of ports per node). All other known graphs fail to reach the Moore limit (indeed, some, like the (3,3) graph in Figure 4, have been proven not to exist) and are thus irregular, which imposes constraints on manufacturability.

In general, the diameter-2 graphs have the following characteristics, where $p$ is the number of ports per node:

- Nodes: $p^2 + 1$
- Ports: $p$
- Bisection Links: $\sim (p-2)(Nodes/4)$
- Maximum Latency: 2

Note that the number of nodes is an upper bound, and only three graphs have been discovered that actually reach this upper bound: the pentagon (effectively the degenerate case), the 10-node Petersen graph, and the 50-node Hoffman-Singleton graph. As one chooses Moore graphs of increasing diameter, the network size achievable with a relatively small number of ports grows rapidly. For instance, on the right of 4 is shown a diameter-3 graph with 22 nodes; a diameter-4 graph has an upper bound of 46. In actuality, the largest known diameter-3 graph has 20 nodes, and the largest known diameter-4 graph has 38. The table below shows the difference between the various bounds (labeled "Max") and the known graph sizes that have been discovered (labeled "Real"): the difference factor grows with both diameter and number of ports [20].

### *Dragonfly and High-Bisection Extensions*

The Dragonfly interconnect [10] is an internet structure, a network of subnetworks. Perhaps the most common form of Dragonfly, which is the form we analyze here, is a fully connected graph of fully connected graphs, which gives it a diameter 3 across the whole network. This is illustrated in Figure 5. Dragonfly networks can use any number of ports for inter-subnet connections, and any number for intra-subnet connections. We vary the number of inter-subnet links, characterizing 1, 2, 4, etc. links connecting each subnet, noting

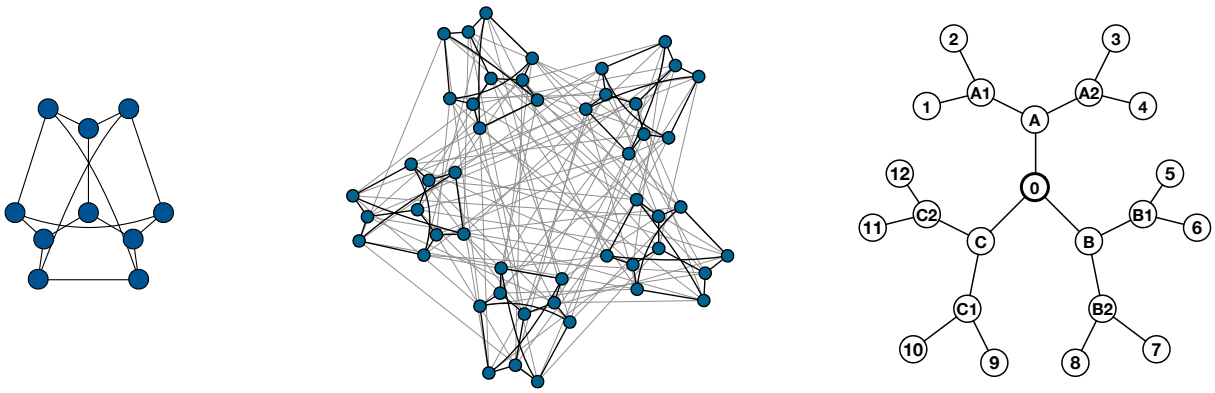


**Figure 4: Two well-known 2-hop Moore graphs: the 10-node Petersen graph (left) and the 50-node Hoffman-Singleton graph (middle); the Moore limit is shown for a 3-hop graph with 3 ports per node (right), without final connecting edges, as the (3,3) limit is unachievable.**

**Table 1: Moore Graphs: Bounds vs. Largest Known**

| | Diameter 2 | | | Diameter 3 | | | Diameter 4 | | |
|---|---|---|---|---|---|---|---|---|---|
| **Ports** | **Max** | **Real** | **Diff** | **Max** | **Real** | **Diff** | **Max** | **Real** | **Diff** |
| **3** | 10 | 10 | = | 22 | 20 | 1.1 | 46 | 38 | 1.2 |
| **4** | 17 | 15 | 1.1 | 53 | 41 | 1.3 | 161 | 96 | 1.7 |
| **5** | 26 | 24 | 1.1 | 106 | 72 | 1.5 | 426 | 210 | 2.0 |
| **6** | 37 | 32 | 1.2 | 187 | 110 | 1.7 | 937 | 390 | 2.4 |
| **7** | 50 | 50 | = | 302 | 168 | 1.8 | 1814 | 672 | 2.7 |
| **8** | 65 | 57 | 1.1 | 457 | 253 | 1.8 | 3201 | 1100 | 2.9 |
| **9** | 82 | 74 | 1.1 | 658 | 585 | 1.1 | 5266 | 1550 | 3.4 |
| **10** | 101 | 91 | 1.1 | 911 | 650 | 1.4 | 8201 | 2286 | 3.6 |
| **11** | 122 | 104 | 1.2 | 1222 | 715 | 1.7 | 12222 | 3200 | 3.8 |
| **12** | 145 | 133 | 1.1 | 1597 | 786 | 2.0 | 17569 | 4680 | 3.8 |
| **13** | 170 | 162 | 1.0 | 2042 | 851 | 2.4 | 24506 | 6560 | 3.7 |
| **14** | 197 | 183 | 1.1 | 2563 | 916 | 2.8 | 33321 | 8200 | 4.1 |
| **15** | 226 | 186 | 1.2 | 3166 | 1215 | 2.6 | 4432 | 11712 | 3.8 |
| **16** | 257 | 198 | 1.3 | 3857 | 1600 | 2.4 | 57857 | 14640 | 4.0 |

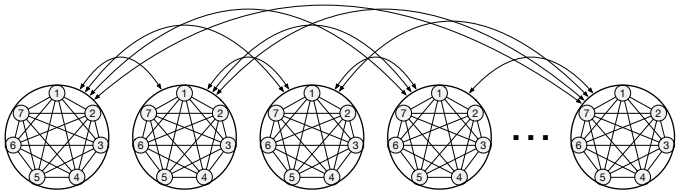

**Figure 5: Dragonfly interconnect: a fully connected graph of fully connected graphs.**

that, when the number of inter-subnet links is equal to one more than the intra-subnet links, the entire network has a diameter of 2, not 3 (if every node has a connection to each of its local nodes as well as a connection to each one of the remote subnets, then it is by definition a two-hop network), which in our graphs later we will label as the "Dragonfly Limit."

In general, Dragonfly networks of this form have the following characteristics, where $p$ is the number of ports for intra-subnet connections, and $s$ is the number of ports connected to remote subnets:

- Nodes: $(p+1)(p+2)$
- Ports: $p+s$
- Bisection Links: $\sim s((p+2)^2/4)$
- Maximum Latency: 3

The bisection bandwidth depends on actual configuration, such as whether the number of subnets $(p+1)$ is even or odd.

### *Fishnet: Angelfish and Flying Fish*

The Fishnet interconnection methodology is a novel means to connect multiple copies of a given subnetwork [8], for instance a 2-hop Moore graph or 2-hop Flattened Butterfly network. Each subnet is connected by multiple links, the originating nodes in each subnet chosen so as to lie at a maximum distance of 1 from all other nodes in the subnet. For instance, in a Moore graph, each node defines such a subset: its nearest neighbors by definition lie at a distance of 1 from all other nodes in the graph, and they lie at a distance of 2 from each other. Figure 6 illustrates.

Using nearest-neighbor subsets to connect the members of different subnetworks to each other produces a system-wide diameter of 4, given diameter-2 subnets: to reach remote subnetwork $i$, one must first reach one of the nearest neighbors of node $i$ within the local subnetwork. By definition,

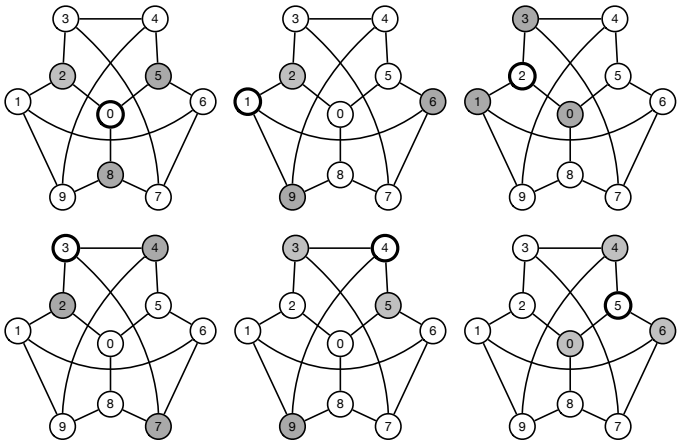

**Figure 6: Each node, via its set of nearest neighbors, defines a unique subset of nodes that lies at a maximum of 1 hop from all other nodes in the graph. In other words, it only takes 1 hop from anywhere in the graph to reach one of the nodes in the subset. Nearest-neighbor subsets are shown in a Petersen graph for six of the graph's nodes.**

this takes at most one hop. Another hop reaches the remote network, where it is at most two hops to reach the desired node. The "Fishnet Lite" variant uses a single link to connect each subnet, as in a typical Dragonfly, and has maximum five hops between any two nodes, as opposed to four.

An example topology using the Petersen graph is illustrated in Figure 7: given a 2-hop subnet of $n$ nodes, each node having $p$ ports (in this case each subnet has 10 nodes, and each node has 3 ports), one can construct a system of $n+1$ subnets, in two ways: the first uses $p+1$ ports per node and has a maximum latency of five hops within the system; the second uses $2p$ ports per node and has a maximum latency of four hops.

The nodes of subnet 0 are labeled $1..n$; the nodes of subnet 1 are labeled $1, 2..n$; the nodes of subnet 2 are labeled $0, 1, 3..n$; the nodes of subnet 3 are labeled $0..2, 4..n$; etc. In the top illustration, node $i$ in subnet $j$ connects directly to node $j$ in subnet $i$. In the bottom illustration, the immediate neighbors of node $i$ in subnet $j$ connect to the immediate neighbors of node $j$ in subnet $i$.

Using the Fishnet interconnection methodology to combine Moore networks produces an Angelfish network, illustrated in Figure 7. Using Fishnet on a Flattened Butterfly network produces a Flying Fish network, illustrated in Figures 8 and 9. Figure 8 illustrates a Flying Fish Lite network based on $7 \times 7 = 49$ -node Flattened Butterfly subnets. The same numbering scheme is used as in the Angelfish example: for all subnets $X$ from 0 to 49 there is a connection between subnet $X$, node $Y$ and subnet $Y$, node $X$. The result is a 2450-node network with a maximum 5-hop latency and 13 ports per node. Note that this is similar to the Cray Cascade [6], in that it is a complete graph of Flattened Butterfly subnets, with a single link connecting each subnet.

Figure 9 gives an example of connecting subnets in a "full" configuration. Fishnet interconnects identify subsets of nodes within each subnetwork that are reachable within a single hop from all other nodes: Flattened Butterflies have numerous such subsets, including horizontal groups, vertical groups, diagonal groups, etc. The example in Figure 9 uses horizontal and vertical groups: 98 subnets, numbered 1H..49H and 1V..49V. When contacting an "H" subnet, one uses any node in the horizontal row containing that numbered node. For example, to communicate from subnet 1H to subnet 16H, one connects to any node in the horizontal row containing node 16. To communicate from subnet 1H to subnet 42V, one connects to any node in the vertical column containing node 42. Given that Flattened Butterfly networks are constructed out of fully connected graphs in both horizontal and vertical dimensions, this means that one can reach a remote subnet in at most two hops. From there, it is a maximum of two hops within the remote subnet to reach the desired target node. For a Flattened Butterfly subnet of $N \times N$ nodes, one can build a system of $2 \times N^4$ nodes with $4 \times N - 2$ ports per node and a maximum latency of 4 hops. This can be extended even further by allowing diagonal sets as well.

In general, the Angelfish graphs have the following characteristics, where $p$ is the number of ports that are used to construct the fundamental Moore graph, from which the rest of the network is constructed. As mentioned above with Moore graphs, the number of nodes is an upper bound, unless specific implementations are described, where the numbers are actual.

### *Angelfish*

- Nodes: $(p^2+1)(p^2+2)$
- Ports: $2p$
- Bisection Links: $\sim p((p^2+1)^2 \div 4)$
- Maximum Latency: 4

### *Angelfish Lite*

- Nodes: $(p^2+1)(p^2+2)$
- Ports: $p+1$
- Bisection Links: $\sim (p^2+1)^2 \div 4$
- Maximum Latency: 5

In general, the Flying Fish graphs have the following characteristics, where $n$ is the length of a side:

### *Flying Fish*

- Nodes: $2n^4$
- Ports: $4n2$
- Bisection Links: $\sim n(n^4) = n^5$
- Maximum Latency: 4

### *Flying Fish Lite*

- Nodes: $n^2(n^2+1)$
- Ports: $2n1$
- Bisection Links: $\sim (n^2+1)^2 \div 4$
- Maximum Latency: 5

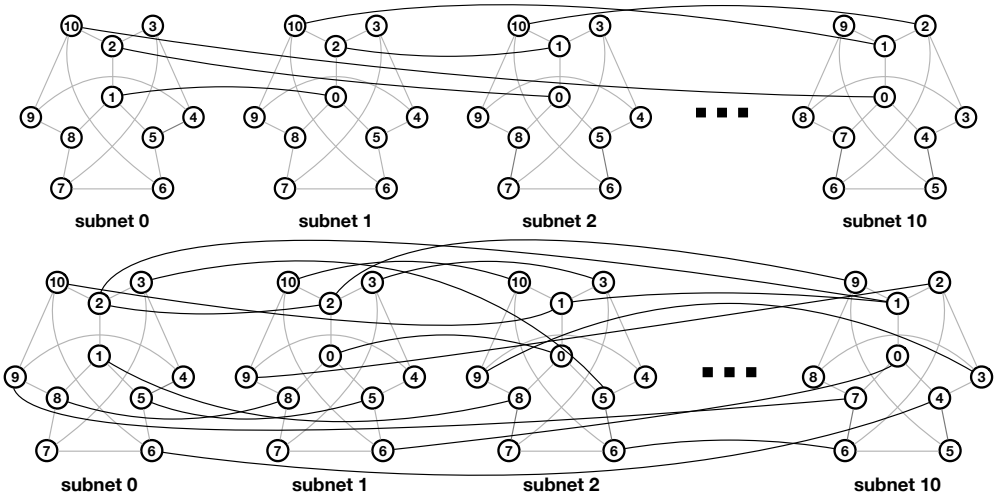


**Figure 7: Angelfish (bottom) and Angelfish Lite (top) networks based on a Petersen graph.**

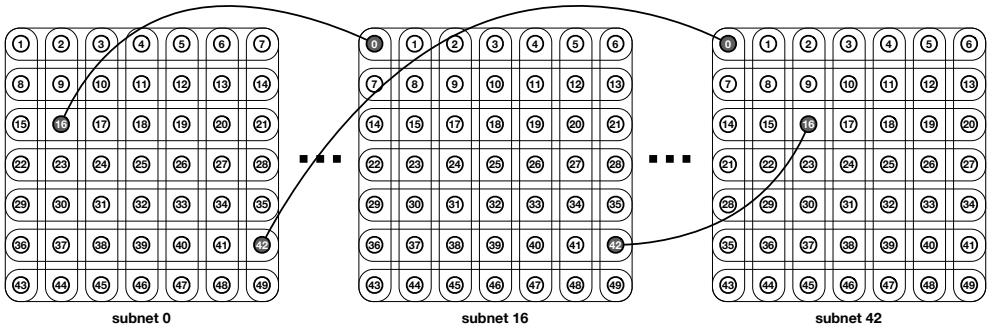


**Figure 8: Flying Fish Lite network based on a 7x7 Flattened Butterfly subnet – 50 subnets of 49 nodes (2450 nodes, 13 ports each, 5-hop latency). Note that this is the same type of arrangement as the Cray Cascade network.**

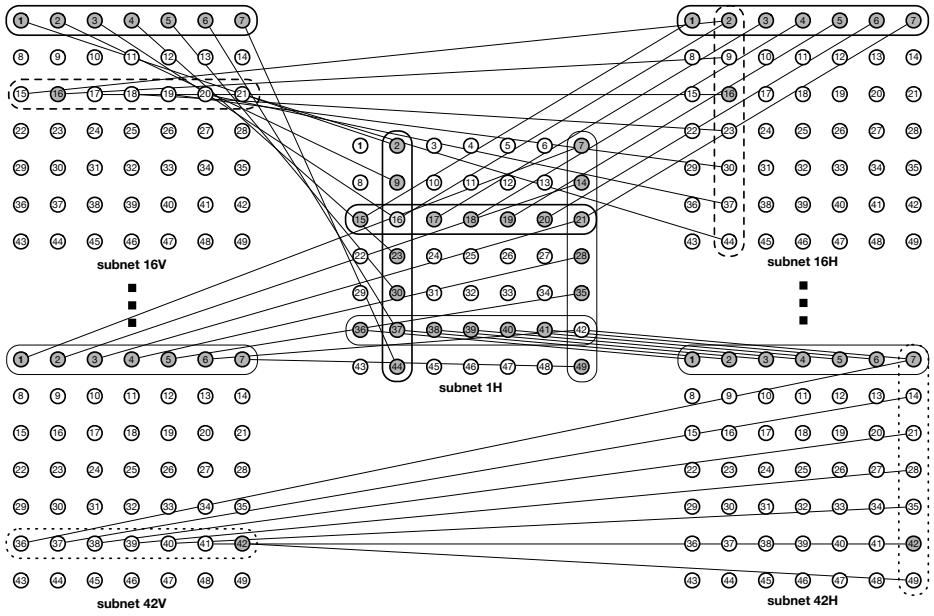


**Figure 9: Flying Fish network based on a 7x7 Flattened Butterfly subnet – 98 subnets of 49 nodes (4802 nodes, 26 ports each, 4-hop latency).**

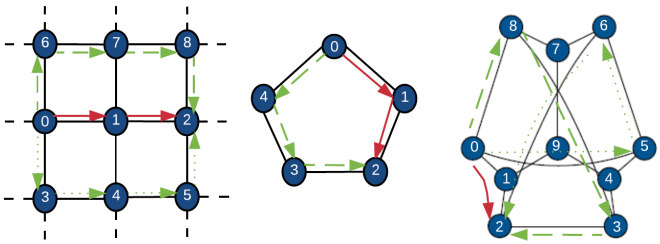


**Figure 10: Rerouting to bypass faulty nodes in 2D torus, 5-node Moore Graph, and a 10-node Petersen Graph. The unavailable route is marked in red; alternative routes marked in green. For simplicity arrows are plotted only in one direction instead of both.**

## 2. NETWORK AVAILABILITY

With millions of nodes in a network, the reliability of the network and the availability of nodes proposes a serious challenge to system designers. A considerable amount of work has been devoted to such issues, from both hardware and software perspectives [15, 22, 5, 19, 14, 18, 1, 9] . As Schroeder [15] pointed out, the failure rates of an HPC system is approximately proportional to the number of nodes (or processors) in that system. Thus for a system building upon millions of nodes, node failure is certain to happen. So in the situation of a node or even a board failure, whether an interconnect topology could provide the ability to reroute to bypass the failed part and allow maintenance and replacement of the failure part is crucial.

Theoretically, all the topologies we have discussed above provide some redundancy in terms of connection – that is, when one node or board/subnet is down, the rest of the network is still able to access other functioning parts of the network. The question remains is, how easily that could be done?

Figure 10 shows the rerouting schemes to bypass a failed node in different topologies. Note that since the nodes of the flattened butterfly are fully connected, it doesn't require rerouting when one node fails; therefore it is not shown in this graph. As could be seen from the graph, assuming in each situation, node 0 is trying to access node 2 through node 1, which is unavailable. A 2D torus has 2 alternative routes to bypass the faulty node, and both take 2 additional hops (4 hops comparing to 2 hops) to get to the destination; a 5-node Moore graph has 1 alternative route, costing 1 additional hop; and a 10-node Petersen graph has 2 alternative routes, costing 1 additional hop. This could be summarized in the following table:

| Topologies for Subnets | Additional Cost in Hops | Alternative Routes |
|---|---|---|
| nD Torus | 2 | $2(n-1)$ |
| Flattened Butterfly | 0 | n/a |
| 5-node Moore | 1 | 1 |
| 10-node Petersen | 1 | 2 |

One interesting result here is that, even though a torus network has more hops to reroute over a failed node than other topologies, it has more alternative routes, especially when the number of dimensions grows. This may imply that during the handling of a failed node in a torus network, there would be more latency, but also more bandwidth and less traffic congestion are expected, which may mitigate the latency issue.

## 3. COST/PERFORMANCE ANALYSIS

This section presents a cost/performance analysis of the previously described network topologies, in configurations up to 1 million nodes. As described above, Moore graphs represent upper bounds; Fishnet numbers based on Moore graphs use actual graphs for a basis (i.e., the Angelfish and Flying Fish numbers in this section are all for real graphs, based on published diameter-2 network topologies, and not unachievable upper bounds on characteristics).

Figure 11 compares the port costs of the topologies with relatively fixed latencies. Instead of having fixed port costs, as in the tori and hypercubes, the port requirements (the number of ports per router) for these topologies grow with the system size, and their average latencies are relatively stable. The Fishnet, Dragonfly, and Flattened Butterfly topologies are compared to the Moore bounds, with Moore graphs of diameter 2, 3, 4, 5, and 6. In general, low-diameter networks can be built out to 1,000,000 nodes with a modest number of ports: high-diameter Flattened Butterfly designs and Fishnet designs will scale to these sizes in under 30 router ports (this is "modest" considering that 100-port routers exist today).

### 3.1 Ports per Router, Average Latencies, Bisection Bandwidth

The graph in Figure 12 shows the average latencies for the topologies that have fixed port costs: the torus variations, from 2D to 10D. A node in a torus interconnect has a fixed number of ports, regardless of the network size, except at 2 nodes per side, which is a hypercube. What changes is the maximum and average latency through the network. One can see that the average latency grows quickly for a system of a given dimension, as the length of a side is scaled from 2 to 3 to 4, etc. One can also see that a torus with 3 nodes on a side is more efficient than a hypercube with the same number of total nodes; this is to be expected, as, given the same dimension, they both have the same worst-case latency, and a torus with 3 nodes per side has more nodes than one with 2 per side.

Figure 13 shows bisection bandwidths; the metric is simply the number of links at the narrowest part of the graph, as each link could be any bandwidth, depending on interconnect technology. The figure in the top left simply plots bisection bandwidth (in links) against system size, both axes logarithm scale. As one would expect, the bandwidths grow proportionally with the system size: as the system scales by four orders of magnitude, the bisection bandwidths scale by over five orders of magnitude. Note that data points are excluded where the design would require more than 300 ports or exhibit an average latency over 35 hops; this is true for all following graphs.

Some of the topologies scale their bisection bandwidth faster than others, and so at the larger system sizes (around 100,000 nodes and larger), there is a clear division of topologies: a gap in bandwidth appears between the top and bottom groups. The top group includes Moore topologies, Dragonfly topologies, Flattened Butterfly topologies, Angelfish, Flying Fish, and Angelfish Mesh. These are the topologies with high port costs, and so it is expected that they would

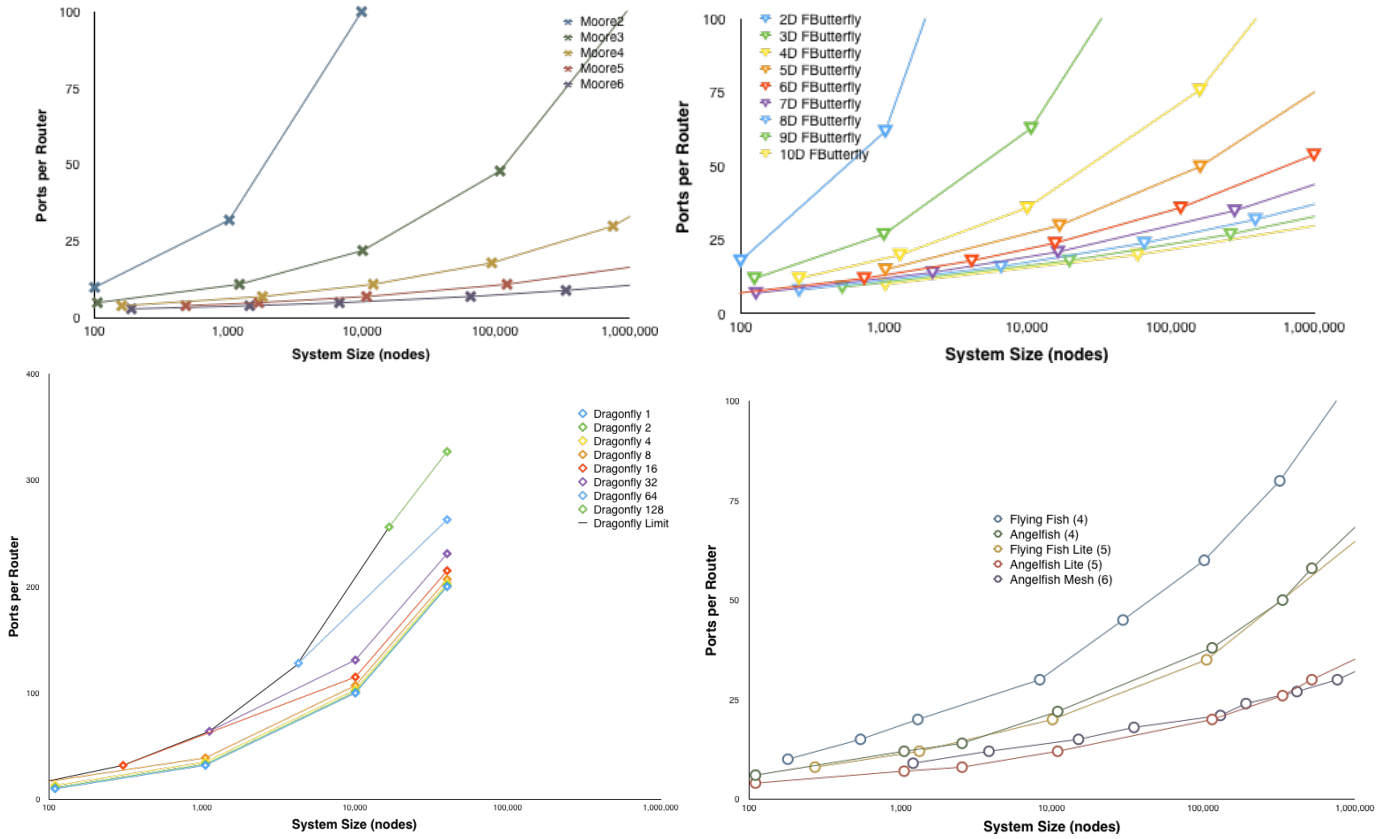


**Figure 11: Ports costs for max-diameter graphs.**

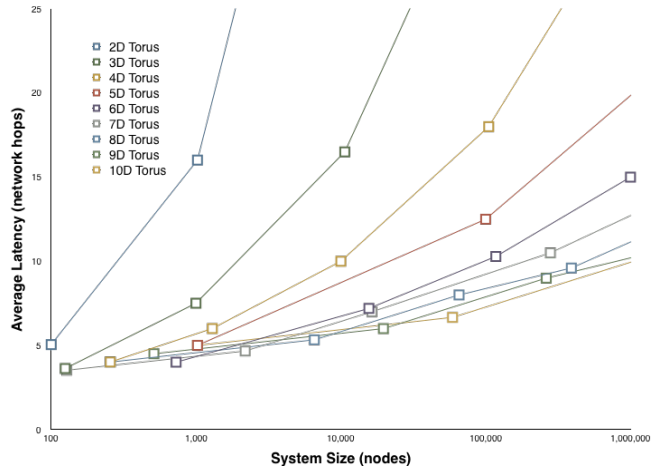


**Figure 12: Average latencies of torus.**

have higher bisection bandwidths. The bottom group includes Angelfish L-Mesh, Angelfish Lite, Flying Fish Lite, and all of the tori from 2D to 10D. The divide can be seen more clearly in the bottom right graph of Figure 13, which presents bisection bandwidth scaled by the system size.

The second graph in Figure 13 shows how efficiently a particular topology can produce bisection bandwidth: this is the ratio of bisection bandwidth to system size. This explains the various curves seen in the previous graph: at the top, including Moore and Flattened Butterfly topologies, Angelfish, Flying Fish, and Angelfish Mesh, the curves grow with system size: i.e., the curves have positive slope, and the bisection bandwidth grows faster than the system size. In addition, topologies such as the Dragonfly variations, Angelfish Lite, and Flying Fish Lite, have designs with completely flat curves, indicating topologies whose bisection bandwidth scales at the same pace as the system size. All of the tori, from 2D to 10D, have downward slopes, indicating that the bisection bandwidth grows more slowly than the system size.

An interesting point is that all topologies other than the tori have lower bandwidth per node as the dimension (and thus the diameter) grows: for instance, the 2-hop Moore and Flattened Butterfly networks are coincident, as are the 3-hop Moore and Flattened Butterfly networks, as are the 4-hop Moore, Flattened Butterfly, and Angelfish networks. Each of these has a shallower slope than the one before it: as dimension and diameter grows, the bisection bandwidth grows less rapidly with the system size. The tori are a mirror image of this: as the dimension grows from 2D to 3D and beyond to 10D, the curves grow flatter in the positive direction, indicating that each has bisection bandwidth that is growing more and more rapidly with system size. They still do not grow as rapidly as the Moore, Flattened Butterfly, and Fishnet topologies, but it is encouraging that tori become better choices the larger their dimension, as the larger dimensions are the only ones that scale to extremely large

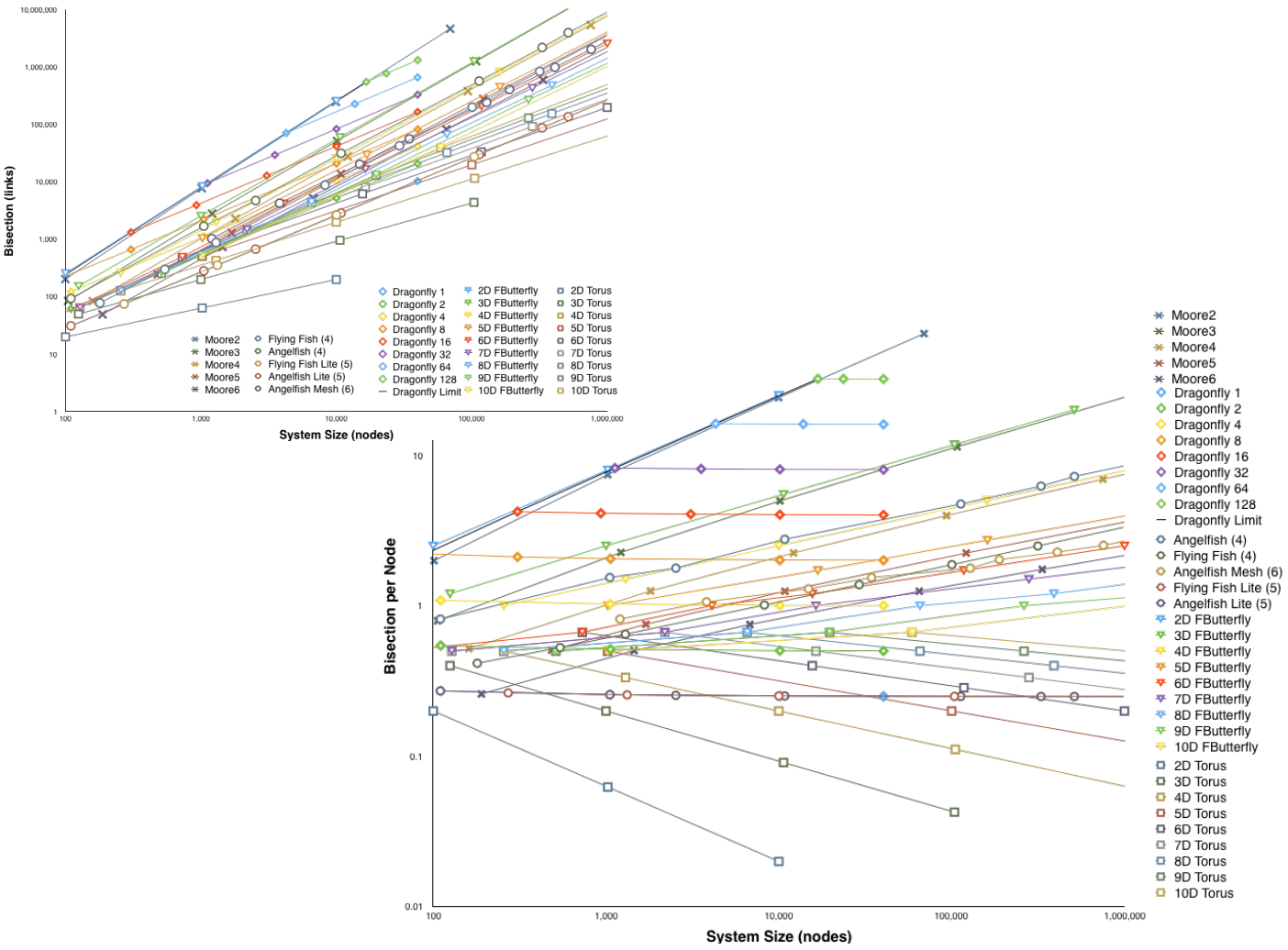


**Figure 13: Left: Bisection bandwidth for all topologies, in terms of links intersected; note the log scale on the y-axis – at nearly all network sizes, there is a two-order-of-magnitude spread in bisection bandwidth, and the spread grows larger with system size. Right: Bisection scaled by system size. This highlights the difference between topologies whose bisection sales faster system size (positive slopes), those that scale more slowly than system size (negative slopes), and those that scale at exactly the same rate (flat slopes).**

network sizes.

## 3.2 Bisection by Ports x Latency

In general, to characterize each system, metrics of interest include *bisection bandwidth*, *average latency*, and the number of *ports per router*, as well as number of nodes. Ideally, one would produce a 3D Pareto surface for each decade of system size to study this design space, but that is difficult to visualize. Instead, we look at a slice through that space by projecting the 3D surface onto one dimension, combining bisection bandwidth, average latency, and ports-per-router cost into one metric. The metric is bisection bandwidth divided by ports and latency; because bisection bandwidth grows much, much faster than the number of ports or the average latency through the network, we scale the metric by the system size. Higher numbers are better.

This metric indicates that, if one interconnect requires twice the number of ports as another, but it has half the latency, the two would be considered equal. If one interconnect has half the average latency as another, but it also has half the bisection bandwidth, the two would be considered equal. If one interconnect has twice the average latency as another, but it only requires half the ports per router, the two would be considered equal. The topologies are compared in Figure 14.

As one can see, there is tremendous variation across the topologies considered. The Moore graphs represent the bound: one cannot achieve a lower port cost at a given diameter, and thus the ports-latency product for Moore graphs is the best achievable. However, some graphs have higher bisection, especially as they use more ports, and so the "efficiency" values can be higher than those of the Moore bounds. A good example of this is the set of diameter-3 Dragonfly variations up near the Dragonfly Limit curve. In this region, the bisec-

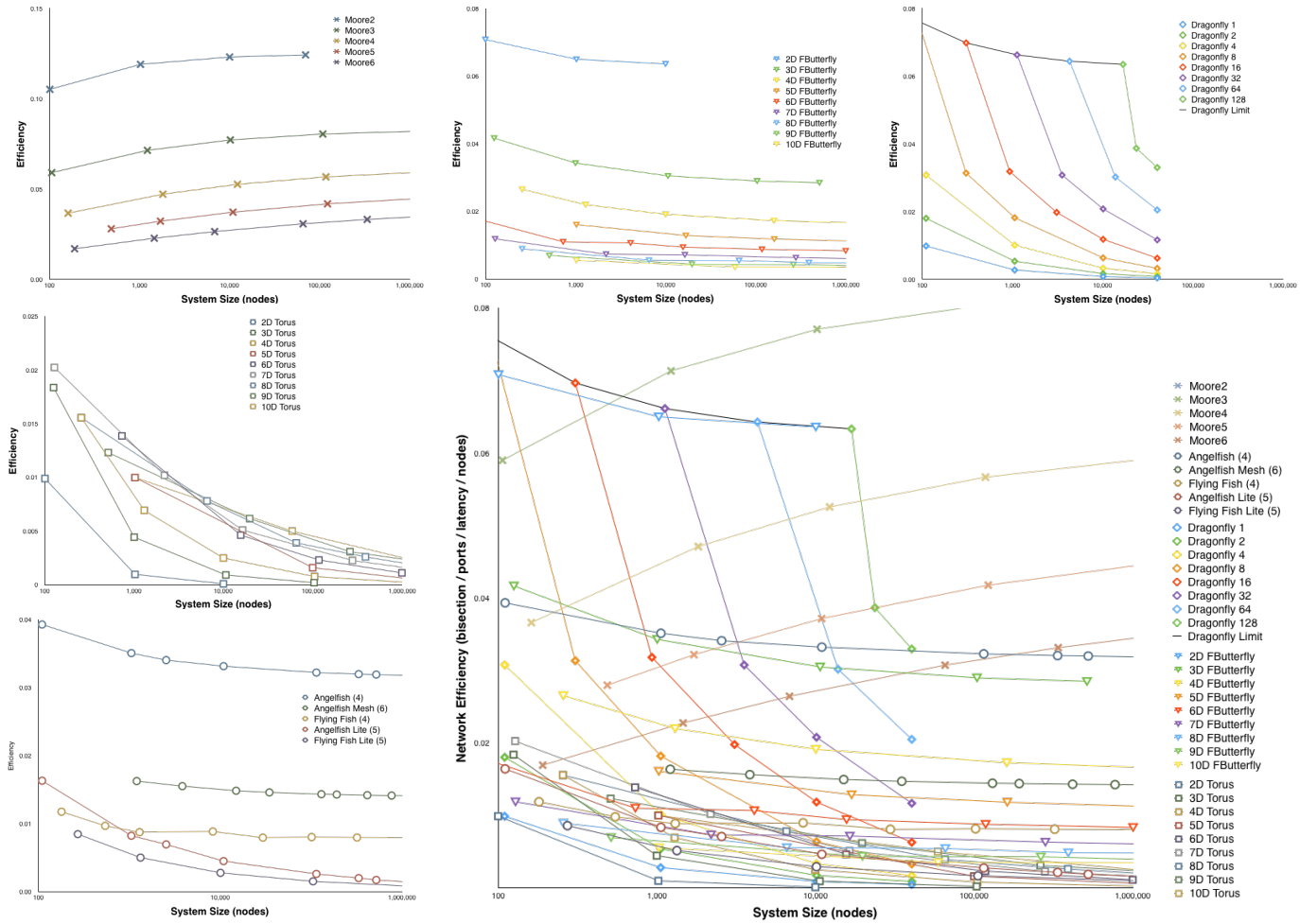


**Figure 14: Bisection per Ports-Latency Product, scaled by System Size. The bottom right graph combines all others into one.**

tion of the network is high because the port costs are high (from right to left, we have Dragonfly networks where an additional 128, 64, 32, etc. links are used to connect each subnetwork, which thereby increases bisection bandwidth by a factor of 128, 64, 32, etc.), and in this region, up to about 1000 nodes, the Dragonfly topologies have higher "efficiency" than the diameter-3 Moore bound.

The Flattened Butterfly designs have very good efficiency, and they scale out to the largest network sizes by increasing dimension and thus diameter. For instance, a diameter-3 Butterfly can be built with 100,000 nodes, requiring 140 ports, and one can scale it to half a million nodes, requiring 240 ports. The design has a high efficiency value because its latency is low and its bisection bandwidth is high.

In the same efficiency neighborhood is a diameter-4 network, the Angelfish design, which scales to 100,000 nodes at 38 ports, to 500,000 nodes at 58 ports, and to 1.1 million nodes at 70 ports. This achieves roughly the same efficiency by having a factor-of-two lower port costs, a factor-of-two lower bisection bandwidth, and 4:3 ratio for average latency. The lower port costs enables it to scale to much larger system sizes for the same port costs.

Thus, out at the largest system sizes, the Fishnet-based topologies (specifically the Angelfish variants) are the most efficient. If one were to follow the curves out far beyond the edge if the graphs shown here, as the network sizes scale larger and larger, the designs that require more ports for larger sizes, all drop out, leaving only the highest-dimensional tori.

## 3.3 Cycle-Accurate Simulations

To compare how the different topologies handle all-to-all traffic, we simulated them using a modified version of Booksim [9] , a widely used, cycle-accurate simulator for interconnect networks. It provides a set of built-in topology models and offers the flexibility for custom topologies by accepting a netlist. The tool uses Dijkstra's algorithm to build the minimum-path routing tables for those configurations that are not in its set of built-in topologies. We simulated injection mode with a uniform traffic pattern. The configurations simulated include the topologies described earlier, as well as 2-hop Moore graphs labeled "MMS2." These latter networks are not bounds but graphs, the same graphs used to construct the Angelfish networks studied in this analysis section; they represent sizes from 18 to 5618 nodes.

The results are shown in Figure 15, which presents aver-

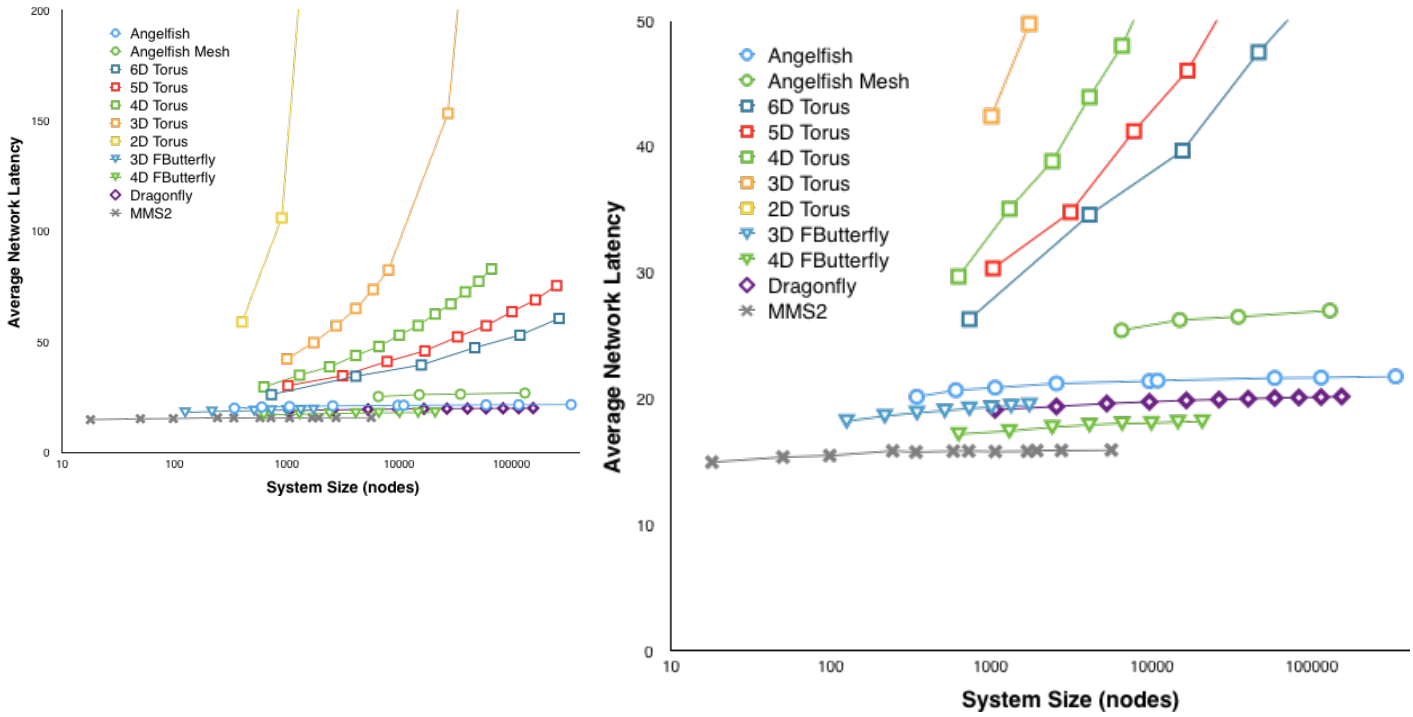


**Figure 15: Simulations of network topologies under constant load; the MMS2 graphs are the 2-hop Moore graphs based on MMS techniques that were used to construct the Angelfish networks.**

age network latency, including transmission time as well as time spent in queues at routers. The figure shows the same graph twice, at different y-axis scales. The left graph shows enough data points to see the sharply increasing slope of the low-dimension tori. The graph on the right shows details of the graphs with the lowest average latencies. There are several things to note. First, it is clear that, at the much higher dimensions, the high-D tori will have latencies on the same scale as the other topologies. Second, the Dragonfly networks are shown scaled out beyond 100,000 nodes, which requires several hundred ports per node, assuming routers are integrated on the CPU. Our simulations show that a configuration using an external router would incur an order of magnitude higher latencies due to congestion at the routers and longer hop counts. The Angelfish and Angelfish Mesh networks at this scale require 38 and 21 ports per node, respectively. Third, the 3D/4D Flattened Butterfly designs have identical physical organization as the 3D/4D tori; they simply use many more wires to connect nodes in each dimension. One can see the net effect: the Flattened Butterfly designs have half the latency of the same-sized tori.

# 4. CONCLUSIONS

There is a clear set of design options to choose from, given the balance between the desire for low average interconnect latency and high bisection bandwidth, and the desire to reduce the number of wires connecting each router chip. Extremely low latencies (e.g., 2 - 3 hops) are certainly possible: two hops can be maintained into the tens of thousands of nodes; three hops can be achieved at system sizes in the range of 500,000 nodes; four hops can be maintained beyond 1 million nodes. The cost for a network is latency and the number of ports per router. If one can live with longer latencies or lower bisection bandwidths, the port requirements can be reduced significantly. On the flip side, if one can live with higher port costs, latencies can be reduced significantly, and bisection bandwidths can be increased significantly. In other words, the "best" topology is not a constant but instead changes with the system size. Router ports can be spent to increase bisection bandwidth, reduce latency (network/graph diameter), and increase total system size: any two can be improved at the expense of the third.

Flattened Butterfly networks match and exceed the bisection bandwidth curves set by Moore bounds and scale well to large sizes by increasing dimension and thus diameter. Dragonfly networks in which the number of inter-group links is scaled have extremely high bisection bandwidth and match that of the Moore bound when extrapolated to their diameter-2 limit. Novel topologies based on Fishnet, in particular the Angelfish design, become efficient at very large sizes (hundreds of thousands of nodes).

One of the most surprising results is that, at the truly extreme scales, high-dimensional tori become the most efficient designs; this, despite the common wisdom that tori do not fare well at large sizes – while that sentiment might hold for 2D and 3D tori, the higher dimensions are extremely powerful designs.

# Acknowledgments

This work was supported in part by the Department of Energy under the *Data Movement Dominates* project, and in part by Northrop Grumman under the *NCTA Cold Logic* program.

## 5. REFERENCES


[1] Y. Ajima, S. Sumimoto, and T. Shimizu. Tofu: A 6d mesh/torus interconnect for exascale computers. *Computer*, 42(11):0036–41, 2009.

[2] W.-T. Bao, B.-Z. Fu, M.-Y. Chen, and L.-X. Zhang. A high-performance and cost-efficient interconnection network for high-density servers. *Journal of computer science and Technology*, 29(2):281–292, 2014.

[3] J. Bermond, J. Bond, M. Paoli, and C. Peyrat. Graphs and interconnection networks: diameter and vulnerability. In *Surveys in combinatorics*, volume 82, pages 1–30. Cambridge University Press Cambridge, 1983.

[4] M. Besta and T. Hoefler. Slim fly: a cost effective low-diameter network topology. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 348–359. IEEE Press, 2014.

[5] F. Cappello, A. Geist, B. Gropp, L. Kale, B. Kramer, and M. Snir. Toward exascale resilience. *International Journal of High Performance Computing Applications*, 2009.

[6] G. Faanes, A. Bataineh, D. Roweth, E. Froese, B. Alverson, T. Johnson, J. Kopnick, M. Higgins, J. Reinhard, et al. Cray cascade: a scalable hpc system based on a dragonfly network. In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, page 103. IEEE Computer Society Press, 2012.

[7] A. J. Hoffman and R. R. Singleton. On moore graphs with diameters 2 and 3. *IBM Journal of Research and Development*, 4(5):497–504, 1960.

[8] B. Jacob. The 2 petaflop, 3 petabyte, 9 tb/s, 90 kw cabinet: A system architecture for exascale and big data.

[9] N. Jiang, J. Balfour, D. U. Becker, B. Towles, W. J. Dally, G. Michelogiannakis, and J. Kim. A detailed and flexible cycle-accurate network-on-chip simulator. In *Performance Analysis of Systems and Software (ISPASS), 2013 IEEE International Symposium on*, pages 86–96. IEEE, 2013.

[10] J. Kim, W. J. Dally, S. Scott, and D. Abts. Technology-driven, highly-scalable dragonfly topology. In *ACM SIGARCH Computer Architecture News*, volume 36, pages 77–88. IEEE Computer Society, 2008.

[11] P. Kogge, K. Bergman, S. Borkar, D. Campbell, W. Carson, W. Dally, M. Denneau, P. Franzon, W. Harrod, K. Hill, et al. Exascale computing study: Technology challenges in achieving exascale systems. 2008.

[12] C. E. Leiserson. Fat-trees: universal networks for hardware-efficient supercomputing. *IEEE transactions on Computers*, 100(10):892–901, 1985.

[13] R. Murphy. On the effects of memory latency and bandwidth on supercomputer application performance. In *2007 IEEE 10th International Symposium on Workload Characterization*, pages 35–43. IEEE, 2007.

[14] A. B. Nagarajan, F. Mueller, C. Engelmann, and S. L. Scott. Proactive fault tolerance for hpc with xen virtualization. In *Proceedings of the 21st annual international conference on Supercomputing*, pages 23–32. ACM, 2007.

[15] B. Schroeder and G. Gibson. A large-scale study of failures in high-performance computing systems. *IEEE Transactions on Dependable and Secure Computing*, 7(4):337–350, 2010.

[16] J. Shalf, S. Dosanjh, and J. Morrison. Exascale computing technology challenges. In *International Conference on High Performance Computing for Computational Science*, pages 1–25. Springer, 2010.

[17] J. S. Smith. *Distributed two-dimensional Fourier Transforms on DSPs: With an applications for phase retrieval*. PhD thesis, 2006.

[18] C. Wang, F. Mueller, C. Engelmann, and S. L. Scott. Proactive process-level live migration in hpc environments. In *Proceedings of the 2008 ACM/IEEE conference on Supercomputing*, page 43. IEEE Press, 2008.

[19] C. Wang, Z. Zhang, X. Ma, S. S. Vazhkudai, and F. Mueller. Improving the availability of supercomputer job input data using temporal replication. *Computer Science-Research and Development*, 23(3-4):149–157, 2009.

[20] Wikipedia. Table of the largest known graphs of a given diameter and maximal degree, 2016. [Online; accessed 27-March-2016].

[21] W. A. Wulf and S. A. McKee. Hitting the memory wall: implications of the obvious. *ACM SIGARCH computer architecture news*, 23(1):20–24, 1995.

[22] J. Yang, D. B. Minturn, and F. Hady. When poll is better than interrupt. In *FAST*, volume 12, pages 3–3, 2012.