High Performance Interconnect Technologies for Supercomputing

Yuanyong Liang, Arnold Galindo, Chike Yuan

High Performance Interconnect Technologies for Supercomputing

Yuanyong Liang*
Illinois Institute of Technology,
Chicago, Illinois, USA

Arnold Galindo*
Illinois Institute of Technology,
Chicago, Illinois, USA

Chike Yuan*
Illinois Institute of Technology,
Chicago, Illinois, USA

ABSTRACT

High-performance computing (HPC) is contributing significantly to scientific progression thanks to its abundant computing power. With recent breakthroughs in natural language processing (NLP), HPC has become indispensable in driving their success, owing to its capacity to train large language models (LLMs). HPC systems not only demonstrate their importance in scientific computing but also in machine learning and generative artificial intelligence. This unprecedented computing power is firstly due to the development of newer computing resources: CPUs, GPUs, FPGAs, etc. However, the progress in interconnect networking, which links hundreds of thousands of compute nodes, should not be overlooked. This survey investigates current popular interconnect topologies driving the most powerful supercomputers.

KEYWORDS

high performance computing, interconnect networking

1 INTRODUCTION

High-Performance Computing (HPC) interconnect topology refers to the arrangement and organization of communication paths in a high-performance computing system. The topology plays a crucial role in determining how different nodes (computers, processors, or other computing elements) within an HPC system communicate with each other. This communication is essential for the efficient processing and transfer of data, which is at the heart of HPC operations [1].

The effectiveness of an HPC interconnect topology is measured by several factors, including its bandwidth (the rate at which data is transferred), latency (the time delay in transferring data), scalability (how well the system performs as more nodes are added), and fault tolerance (the system's ability to continue operating in the event of a failure) [2].

There are several types of HPC interconnect topologies, such as fat-tree [3], dragonfly [4, 5], and HyperX [6]. Each topology is designed to serve special purposes with its own set of advantages and disadvantages. The choice of topology depends on the specific requirements and goals of the HPC system, such as the type of computations it will perform, the size of the datasets, budget constraints, and the desired balance between speed, cost, and reliability.

In recent years, there has been significant innovation in HPC interconnect technologies, driven by the growing demands for faster and more efficient computing capabilities, especially in fields like climate modeling, genomic analysis, and artificial intelligence[7–10]. These advances continue to push the boundaries of what's possible in high-performance computing.

*All authors contributed equally to this work.

2 FAT-TREE TOPOLOGY

Fat-tree topology is a hierarchical network design that mimics a tree structure, where each layer in the hierarchy is connected upward to the next layer until it reaches the root [3, 11]. This topology is widely used in large-scale and high-performance computing environments, such as data centers and supercomputers, due to its scalability, high bandwidth, and fault tolerance capabilities. In the following sections, we will discuss the architecture of fat-tree topology, its advantages, challenges, and applications.

2.1 Architecture

The architecture of a fat-tree topology is designed to overcome the limitations of traditional tree topologies, especially in terms of bandwidth and scalability. In a fat-tree, the concept of a "fat" tree is realized by increasing the number of links and the bandwidth as one moves up the hierarchy. This is in contrast to a conventional tree where all links typically have the same bandwidth. As depicted in Figure 1, the bottom layer of a fat-tree consists of edge switches, which connect to nodes such as servers or computers. The next layer consists of aggregation switches, which connect multiple edge switches. Finally, at the top of the hierarchy, core switches interconnect the aggregation switches. The key feature of this architecture is that the capacity of the switches and the number of links between layers increase as one moves up the hierarchy, ensuring that the network can handle high volumes of traffic without bottlenecks.

2.2 Routing

Static routing is majorly deployed on fat-tree systems. Static routing forwards packets in a deterministic manner such that the packets between the same source-destination node pair always follow the same routing path. The "destination modulo k" (D-mod-k) algorithm is a widely adopted static routing technique that effectively distributes traffic across links in a fat-tree topology, demonstrating strong performance characteristics [12]. This method determines the next upward link at each hierarchical level by utilizing the destination node's ID, continuing this process until it reaches the shared ancestor node. Subsequently, the algorithm selects the appropriate downward links to direct the traffic towards the intended destination.

In contrast to static routing, adaptive routing forwards packets based on real-time network estimation, thus packets from the same message can be delivered using different paths [13, 14]. In addition to static routing, several works also bring adaptive routing to fat-tree topologies [15, 16]. Adaptive routing allows packets between a source-destination node pair to follow different forwarding paths. While routing flexibility is improved, special designs for choosing the best path and handling out-of-order packet delivery need to be implemented.

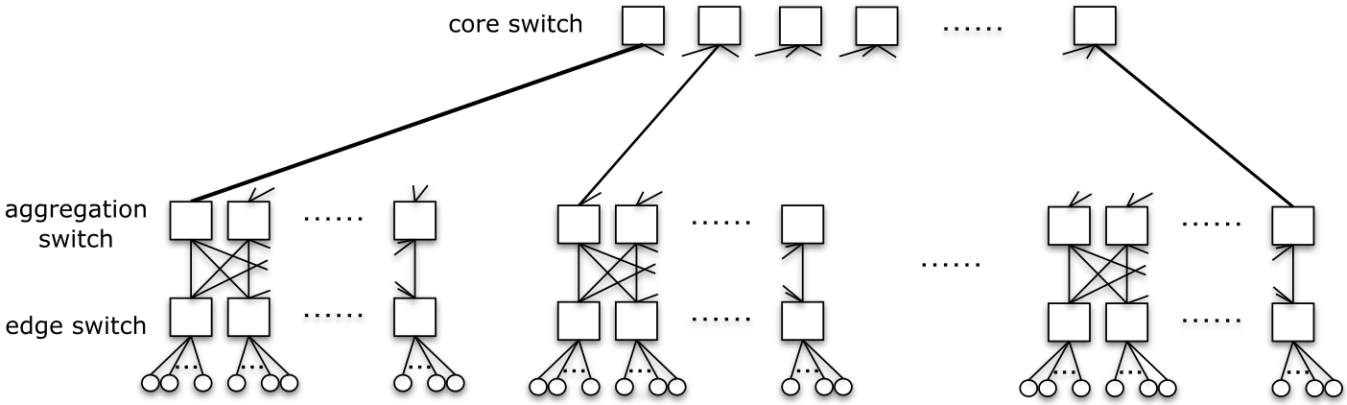

Figure 1: An example of a three-level fat-tree

2.3 Discussions

Fat-tree topology is a popular choice for HPC and large data center networks due to is hierarchical design. However, there are several points that need to be carefully considered for deploying fat-tree. Especially the ones discussed below.

- Scalability: One of the most significant advantages of fat-tree topology is its scalability. Due to its hierarchical nature and the increased bandwidth at higher layers, fat-trees can easily accommodate growth in network size by adding more nodes at the edge or enhancing the capacity of the upper layers.
- High Bandwidth: The increased link capacity and parallel paths available in fat-tree topology allow for high bandwidth between any two nodes in the network. This is particularly beneficial for applications requiring significant data transfer, such as cloud computing and big data analytics.
- Fault Tolerance: Fat-tree topology offers inherent fault tolerance through multiple redundant paths between any two points in the network. If a link or switch fails, the network can dynamically reroute traffic, minimizing downtime and ensuring continuous availability.
- Complexity: The design and implementation of a fat-tree topology can be complex, especially as the network scales. Managing the numerous switches and potential paths requires sophisticated network management tools and algorithms to ensure efficient traffic flow and fault management.
- Cost: The initial setup cost of a fat-tree topology can be high due to the need for switches with higher bandwidth capacities, especially at the upper layers of the hierarchy. Operational costs can also increase with the need for advanced network management and monitoring tools.

Fat-tree topology offers a scalable, high-bandwidth, and fault-tolerant network architecture that is well-suited to the demands of modern high-performance computing environments. While it presents challenges in terms of complexity and cost, the benefits it provides in terms of performance and reliability make it an attractive choice for data centers, cloud computing infrastructure, and supercomputing.

3 DRAGONFLY TOPOLOGY

Dragonfly is one of the most popular topologies used in the design of high-performance computing networks and supercomputers. The topology demonstrates its innovation and efficiency by deploying on the two most powerful HPC systems in the Top 500 November list[17].

3.1 Architecture

Dragonfly is specifically designed to address the challenges of scalability, low latency, and high bandwidth requirements from nowadays large-scale workloads and applications. The name "dragonfly" is inspired by the natural structure of a dragonfly's wings, which symbolizes the topology's flexibility to connect multiple groups of nodes for different system scales.

As shown in Figure 2, the dragonfly topology is a hierarchical network architecture that consists of groups of routers. These groups are interconnected in a way that minimizes the number of hops data must travel between any two points in the network. This design significantly reduces latency and increases the overall efficiency of data transmission [4]. Each group in a dragonfly network contains a set of routers, and each router is connected to routers within the same group and to routers in other groups using local channels and global channels respectively. This creates a high level of redundancy and paths for data to travel, enhancing the network's fault tolerance and reliability. Many flavors of dragonfly have been deployed depending on their intra-group connection. For example, the current Slingshot network deployed on Frontier and Aurora supercomputer constructs a one-dimensional full intra-group network[18][19][20]. Aries network arranges routers in a group as a two one-dimensional grid, and each dimension is fully connected[21]. Dragonfly+ constructs routers into a tree-like bipartite graph intra-group connection[22][23].

3.2 Advantages of Dragonfly Topology

One of the primary advantages of the dragonfly topology is its scalability. As computational demands grow, networks designed with dragonfly topology can be expanded by adding more groups or increasing the number of routers within groups, without significantly impacting the network's performance. This makes it an ideal choice

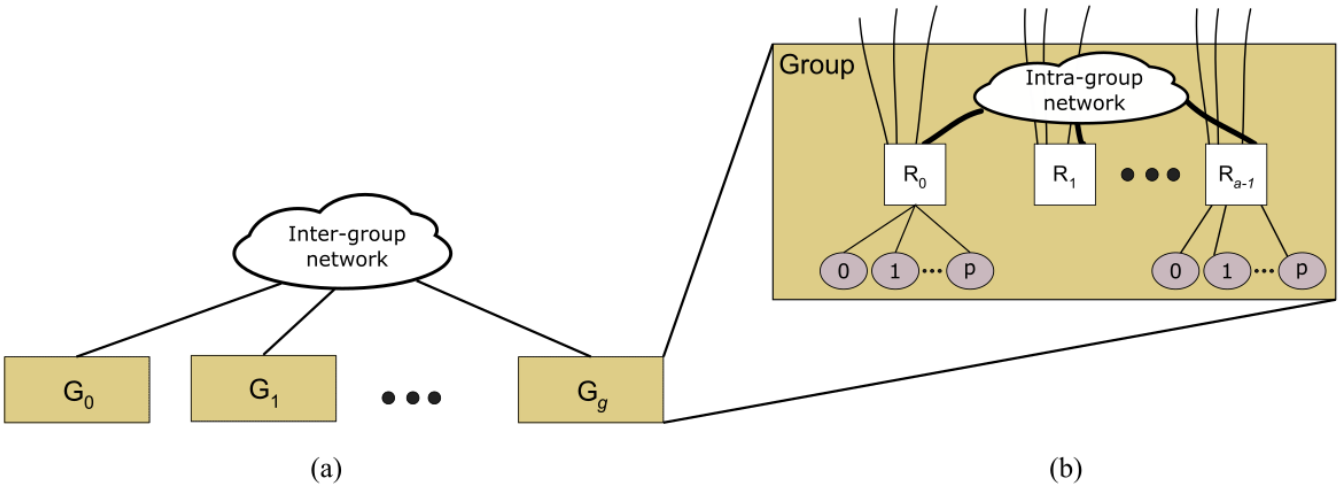

Figure 2: Hierarchical connections in dragonfly topology. (a) groups are all-to-all connected through the inter-group network with global channels. (b) routers within a group are linked through local channels and connected to the terminals through terminal channels.

for supercomputing environments where computational needs are continuously evolving.

Another significant advantage is the high bandwidth and low latency guaranteed by different dragonfly flavors[24, 25]. By minimizing the number of hops between nodes, the dragonfly topology ensures that data packets reach their destination more quickly compared to traditional network designs. This is particularly beneficial in applications that require real-time data processing and transmission, such as scientific simulations and data analytics.

3.3 Challenges and Considerations

While the dragonfly topology offers numerous benefits, it also presents certain challenges. The complexity of the network's design can make it difficult to implement and manage, especially in very large-scale systems. Additionally, the initial cost of setting up a dragonfly network can be higher than traditional networks due to the specialized hardware required. Another challenge in network design that should not be overlooked is the network workload interference[26, 27], which usually requires additional QoS or congestion control mechanisms to provide a higher network capability [20, 28, 29]. However, these challenges are often outweighed by the long-term benefits of improved performance and scalability.

In conclusion, the dragonfly topology represents a significant advancement in the design of network architectures for high-performance computing environments. Its ability to provide low latency, high bandwidth, and exceptional scalability makes it an ideal choice for supercomputers and large-scale data centers. Despite its challenges, the dragonfly topology continues to gain popularity as the demand for faster and more efficient computing networks grows.

4 HYPERX TOPOLOGY

HyperX topology is a significant concept in the field of network architecture, particularly relevant to the design of high-performance computing (HPC) systems and data centers [6]. Unlike the more commonly known dragonfly topology, which is characterized by its hierarchical structure, HyperX is recognized for its flexibility and scalability, offering an alternative approach to constructing large-scale, high-bandwidth, and low-latency networks[30].

4.1 Architecture

HyperX topology is designed to be a highly scalable and efficient network architecture that can support a vast number of nodes with relatively low latency and high bandwidth. It is a generalization of the flattened butterfly and flattened torus topologies, allowing for more flexibility in network design and scalability. The HyperX topology constructs high-dimensional interconnection networks by crossing multiple layers of the network, which can be tailored to the specific needs and scale of a computing environment.

The key characteristic of HyperX topology is its multi-dimensional mesh structure, which allows for multiple paths between any two nodes in the network. In HyperX, routers on the same dimension are fully connected. For example, figure3 illustrates the link connection of two HyperX networks at different dimensionality. Figure3(a) shows a two-dimensional HyperX with two routers in the first dimension and four routers in the second. Figure3(b) depicts a three-dimensional HyperX, with three routers on each dimension. The redundancy in paths between two nodes enhances the network's fault tolerance and load balancing capabilities, as traffic can be rerouted in the event of a node or link failure, ensuring consistent performance and reliability.

4.2 Advantages of HyperX Topology

One of the primary advantages of the HyperX topology is its scalability. As computational requirements grow, networks designed with HyperX can easily be expanded by adding more dimensions or nodes within each dimension. This flexibility makes it an attractive option for evolving HPC environments and data centers that

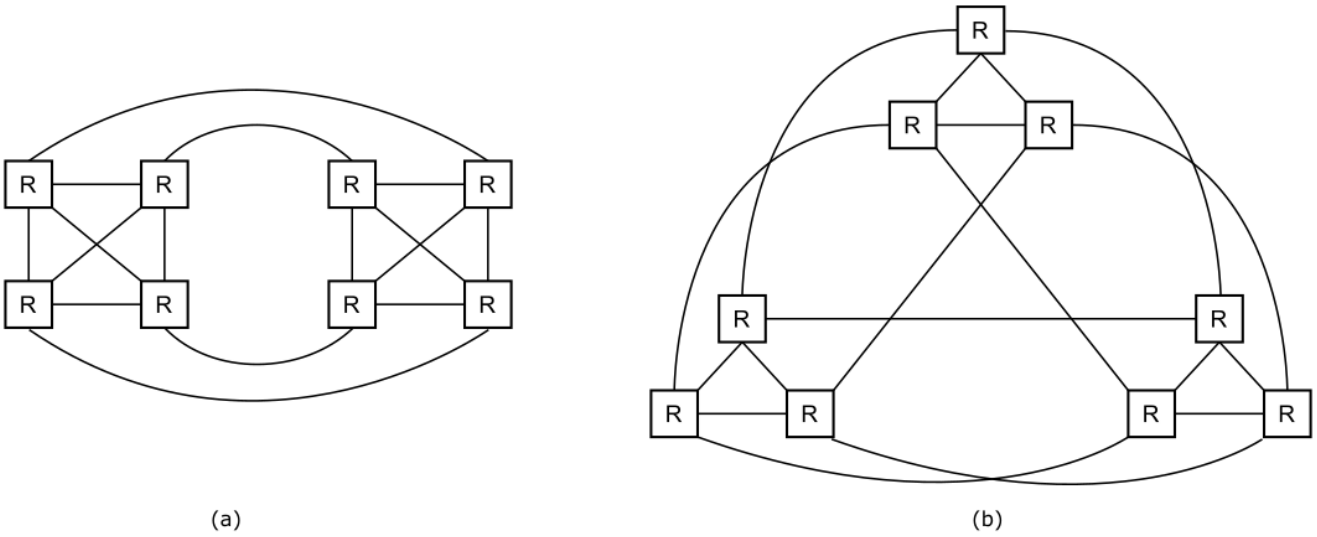

Figure 3: Link connections between routers in the HyperX topology, nodes are omitted. (a) Two-dimensional HyperX with two and four routers on each dimension. (b) Three-dimensional HyperX, with three routers on each dimension.

need to accommodate increasing data volumes and computational demands.

Moreover, HyperX offers high bandwidth and low latency by providing direct paths between nodes and reducing the number of hops required for data transmission[31]. This is particularly beneficial for applications that require rapid communication between nodes, such as parallel computing tasks and real-time data processing.

4.3 Challenges and Considerations

Despite its advantages, implementing HyperX topology comes with its set of challenges [32, 33]. Designing and managing a multi-dimensional network can be complex, requiring advanced planning and specialized knowledge to optimize performance and ensure reliability [34]. Additionally, the initial setup and hardware costs can be higher compared to traditional network topologies, making it a significant investment.

In conclusion, HyperX topology represents a powerful and flexible approach to designing network architectures for high-performance computing environments. Its scalability, high bandwidth, and low latency make it an ideal choice for a wide range of applications, from scientific simulations to cloud computing. While there are challenges in implementing HyperX topology, its potential benefits in terms of performance and adaptability make it a compelling option for future network designs in HPC systems and beyond. As computational demands continue to grow, the importance of efficient and scalable network topologies like HyperX will undoubtedly increase, driving further innovation in this field.

5 CONCLUSION

In this paper, we analyzed three popular interconnect topologies that are widely adopted in HPC and data center large-scale parallel systems. The studied topologies are fat-tree, dragonfly, and HyperX networks. All these three topologies show their advantages in supporting extreme-scale systems with high bandwidth, low latency, and flexibility to support systems at different scale.

REFERENCES

[1] Dennis Abts and John Kim. *High performance datacenter networks: Architectures, algorithms, and opportunities*. Morgan & Claypool Publishers, 2011.

[2] William James Dally and Brian Patrick Towles. *Principles and practices of interconnection networks*. Elsevier, 2004.

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

[4] John Kim, Wiliam J Dally, Steve Scott, and Dennis Abts. Technology-driven, highly-scalable dragonfly topology. *ACM SIGARCH Computer Architecture News*, 36(3):77–88, 2008.

[5] Yao Kang, Xin Wang, and Zhiling Lan. Study of workload interference with intelligent routing on dragonfly. In *SC22: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–14. IEEE, 2022.

[6] Jung Ho Ahn, Nathan Binkert, Al Davis, Moray McLaren, and Robert S Schreiber. Hyperx: topology, routing, and packaging of efficient large-scale networks. In *Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis*, pages 1–11, 2009.

[7] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[8] Xin Wang, Misbah Mubarak, Yao Kang, Robert B Ross, and Zhiling Lan. Union: An automatic workload manager for accelerating network simulation. In *2020 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 821–830. IEEE, 2020.

[9] David Van Der Spoel, Erik Lindahl, Berk Hess, Gerrit Groenhof, Alan E Mark, and Herman JC Berendsen. Gromacs: fast, flexible, and free. *Journal of computational chemistry*, 26(16):1701–1718, 2005.

[10] James C Phillips, Rosemary Braun, Wei Wang, James Gumbart, Emad Tajkhorshid, Elizabeth Villa, Christophe Chipot, Robert D Skeel, Laxmikant Kale, and Klaus Schulten. Scalable molecular dynamics with namd. *Journal of computational chemistry*, 26(16):1781–1802, 2005.

[11] Fabrizio Petrini and Marco Vanneschi. k-ary n-trees: High performance networks for massively parallel architectures. In *Proceedings 11th international parallel processing symposium*, pages 87–93. IEEE, 1997.

[12] Crispín Gomez, Francisco Gilabert, María Engracia Gomez, Pedro López, and José Duato. Deterministic versus adaptive routing in fat-trees. In *2007 IEEE International Parallel and Distributed Processing Symposium*, pages 1–8. IEEE, 2007.

[13] Yao Kang, Xin Wang, and Zhiling Lan. Q-adaptive: A multi-agent reinforcement learning based routing on dragonfly network. In *Proceedings of the 30th International Symposium on High-Performance Parallel and Distributed Computing*, pages 189–200, 2021.

[14] Nan Jiang, John Kim, and William J Dally. Indirect adaptive routing on large scale interconnection networks. In *Proceedings of the 36th annual international symposium on Computer architecture*, pages 220–231, 2009.

[15] Staci A Smith, Clara E Cromey, David K Lowenthal, Jens Domke, Nikhil Jain, Jayaraman J Thiagarajan, and Abhinav Bhatele. Mitigating inter-job interference using adaptive flow-aware routing. In *SC18: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 346–360. IEEE, 2018.

[16] José Rocher-González, Ernst Gunnar Gran, Sven-Arne Reinemo, Tor Skeie, Jesús Escudero-Sahuquillo, Pedro Javier García, and Francisco J. Quiles Flor. Adaptive routing in infiniband hardware. In *2022 22nd IEEE International Symposium on Cluster, Cloud and Internet Computing (CCGrid)*, pages 463–472, 2022.

[17] Top500 list. https://www.top500.org/lists/top500/2023/11/. Accessed: 2023-12-11.

[18] Aurora supercomputer. https://www.alcf.anl.gov/aurora. Accessed: 2023-12-11.

[19] Frontier supercomputer. https://www.olcf.ornl.gov/frontier/. Accessed: 2023-12-11.

[20] Daniele De Sensi, Salvatore Di Girolamo, Kim H McMahon, Duncan Roweth, and Torsten Hoefler. An in-depth analysis of the slingshot interconnect. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–14. IEEE, 2020.

[21] Bob Alverson, Edwin Froese, Larry Kaplan, and Duncan Roweth. Cray xc series network. *Cray Inc., White Paper WP-Aries01-1112*, 2012.

[22] Alexander Shpiner, Zachy Haramaty, Saar Eliad, Vladimir Zdornov, Barak Gafni, and Eitan Zahavi. Dragonfly+: Low cost topology for scaling datacenters. In *2017 IEEE 3rd International Workshop on High-Performance Interconnection Networks in the Exascale and Big-Data Era (HiPINEB)*, pages 1–8. IEEE, 2017.

[23] Mario Flajslik, Eric Borch, and Mike A Parker. Megafly: A topology for exascale systems. In *High Performance Computing: 33rd International Conference, ISC High Performance 2018, Frankfurt, Germany, June 24-28, 2018, Proceedings 33*, pages 289–310. Springer, 2018.

[24] Yao Kang, Xin Wang, Neil McGlohon, Misbah Mubarak, Sudheer Chunduri, and Zhiling Lan. Modeling and analysis of application interference on dragonfly+. In *Proceedings of the 2019 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*, pages 161–172, 2019.

[25] Majid Salimi Beni and Biagio Cosenza. An analysis of long-tailed network latency distribution and background traffic on dragonfly+. In *International Symposium on Benchmarking, Measuring and Optimization*, pages 123–142. Springer, 2022.

[26] Yao Kang. *Workload Interference Analysis and Mitigation on Dragonfly Class Networks*. PhD thesis, Illinois Institute of Technology, 2022.

[27] Xing Pu, Ling Liu, Yiduo Mei, Sankaran Sivathanu, Younggyun Koh, and Calton Pu. Understanding performance interference of i/o workload in virtualized cloud environments. In *2010 IEEE 3rd International Conference on Cloud Computing*, pages 51–58. IEEE, 2010.

[28] Jeremiah J Wilke and Joseph P Kenny. Opportunities and limitations of quality-of-service in message passing applications on adaptively routed dragonfly and fat tree networks. In *2020 IEEE International Conference on Cluster Computing (CLUSTER)*, pages 109–118. IEEE, 2020.

[29] Marina García, Enrique Vallejo, Ramón Beivide, Mateo Valero, and Germán Rodríguez. Ofar-cm: Efficient dragonfly networks with simple congestion management. In *2013 IEEE 21st Annual Symposium on High-Performance Interconnects*, pages 55–62. IEEE, 2013.

[30] Sadoon Azizi, Farshad Safaei, and Naser Hashemi. On the topological properties of hyperx. *The Journal of Supercomputing*, 66:572–593, 2013.

[31] Georgios Kathareios, Cyriel Minkenberg, Bogdan Prisacari, German Rodriguez, and Torsten Hoefler. Cost-effective diameter-two topologies: Analysis and evaluation. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–11, 2015.

[32] Jens Domke, Satoshi Matsuoka, Ivan Radanov, Yuki Tsushima, Tomoya Yuki, Akihiro Nomura, Shin'ichi Miura, Nic McDonald, Dennis Lee Floyd, and Nicolas Dubé. The first supercomputer with hyperx topology: A viable alternative to fat-trees? In *2019 IEEE Symposium on High-Performance Interconnects (HOTI)*, pages 1–4. IEEE, 2019.

[33] Yao Kang, Xin Wang, and Zhiling Lan. Workload interference prevention with intelligent routing and flexible job placement on dragonfly. In *Proceedings of the 2023 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*, pages 23–33, 2023.

[34] Jayaram Mudigonda, Praveen Yalagandula, and Jeffrey C Mogul. Taming the flying cable monster: A topology design and optimization framework for {Data-Center} networks. In *2011 USENIX Annual Technical Conference (USENIX ATC 11)*, 2011.