Noname manuscript No.
(will be inserted by the editor)

Optimal Routing for a Family of Scalable Interconnection Networks

Zhipeng Xu · Xiaolong Huang · Yuefan Deng

Received: date / Accepted: date

Abstract Scalability of interconnection networks for the supercomputers, particularly, for next-generation exascale systems with tens of millions of cores, is one of the essential features for enhancement of performance. One of the required properties of the scalable and low-radix for interconnection networks is the minimum-diameter-based scalable (MDBs) network. The MDBs network combines the desired features of some optimal topologies with different orders, such as small diameter, high connectivity, symmetry, and regularity. The diameter of the MDBs network grows up linearly while the size of the network measured in nodes increases exponentially. We designed the vertex-balanced routing algorithm for the base network by considering the pressure of the data transit in each node. The benchmarks on a real small-scale cluster show amazing improvements in performance after adapting the new routing algorithm. Each node of MDBs generated from base topologies can also sustain balanced transit loading if we apply the optimized routing algorithm to the base network. The simulation results show that our algorithm can substantially enhance communication performance for MDBs.

Keywords Interconnection network · optimal routing

Zhipeng Xu
1. School of Data and Computer Science
Sun Yat-sen University, Guangzhou, Guangdong 510275, P.R. China
2. Department of Applied Mathematics and Statistics
Stony Brook University, Stony Brook, NY 11794, USA
E-mail: xzpmail@gmail.com

Xiaolong Huang
Department of Applied Mathematics and Statistics
Stony Brook University, Stony Brook, NY 11794, USA
E-mail: xiaolong.huang@stonybrook.edu

Yuefan Deng (Corresponding author)
Department of Applied Mathematics and Statistics
Stony Brook University, Stony Brook, NY 11794, USA
E-mail: yuefan.deng@stonybrook.edu

1 Introduction

The performance of supercomputers has been advancing sustainably for the past few decades, and the exascale system will be realized in next-generation shortly. The speed of a single processor approaches a hard wall and Moore's [37] law is slowing down, caused by the limitation of physics. To pursue extreme speed, modern supercomputers in TOP500 [1] always contain millions of cores, and particularly, the number of cores for exascale supercomputers will rise to 10 million to 100 million, which is a challenge of efficiency and scalability for interconnection network. The n-cube binary, also called hypercube [15, 21], along with tours [3,4,26], fat-tree [30,31], and Dragonfly [25,49], has also been widely implemented and proved to be very powerful topologies. Among these topologies, hypercube and tours are direct networks, which is similar to the peer-to-peer network, the messages can be exchanged directly between systems on the network without external devices, e.g., server, switcher. On the contrary, fat-tree and the Dragonfly are classical indirect networks, but the boundary between direct networks and indirect networks is not clear because every network can be considered as an indirect network after dividing the computing node into computing unit and switching unit [23].

For designing the interconnection network, the diameter is a significant parameter because it is related to communication latency for topology. To further reduce the diameter, there exist two directions roughly; the first one is to search directly, and the other one is to combine two or more base graphs. The degree/diameter [35] problem has relevance to design the network directly, which aims at discovering the largest possible graph G of diameter k such that the largest degree any of the vertices in G is at most d. In particular, to promote degree/diameter problem and design the topology for future supercomputers, the Graph Golf [28], an international competition, is conducted to find a series of minium-diameter graphs for every order/degree pair. However, the heuristic method [27] or random search for graphs with larger orders would cost massive computing resources, that may not be accepted in engineering. Hence, the hierarchical networks have been proposed to construct larger networks easily and meet scalability and modularity [32,33], which are essential requirements for a multi-computer system and become mainstream topologies of most supercomputers. Dragonfly is a type of classical hierarchical network, dividing the links into intra-group and inter-group, which only to need to optimize the links part by part, while the essence of fat-tree is a multi-level multistage interconnection network (MIN) [52]. Hypercube, cross-cube [20], folded-Heawood [24,2], hyper-Petersen, and folded-Petersen [12] are a class of product networks, which using CARTESIAN product [13,54,56] operation to combine some graphs and can be considered as a special class of hierarchical networks contain two or more product items. To reduce the number of links between base graphs, a class of recursive networks, extended hypercubes (EH) [29], hierarchical Petersen network (HPN) [43], and recursive expand Heawood (REH)[51] have also been proposed. Besides, block shift network (BSN) [41], hierarchical folded-hypercube network (HFN) [45], hierarchical star (HS) [44], hierarchical hypercube network (HHN) [34] and Dandelion [17], etc., are also novel with their own strengths, that can be applied to supercomputers and data centers.

However, most hierarchical networks are based on the Cube graph, Peterson graph, Heawood graph, and HoffmanSingleton graph [19] of order 4, 10, 14, 50, which lead to the large step of vertices number without fine-tuning for a large machine, due to its lack of scalability. With combing the discoveries for optimal graphs of order 16, 32, we proposed the minimum-diameter-based scalable (MDBs) network, a family of scalable and low-radix for interconnection networks. The MDBs use some base graphs with minimum diameters to create a larger network by using product operation but without the limitation of orders for base graphs.

In the interconnection network, the routing algorithm is an indispensable part, that is relevant to the efficiency of networks; A bad routing would limit the communication capacity due to bottleneck, though with the optimal graph. Usually, the routing algorithms can be classified as deterministic, oblivious, and adaptive according to how to select one path from a multi-path set of two nodes [23]. In practice, if the algorithms have been designed only with the rule of short path length but not considering balance the load evenly over all channels, that would cut throughput in theory. Actually, the shortest path length is just a second important aspect of any routing algorithm [23]. Valiant's algorithm [50] is a classical method by inserting an intermediate node randomly to balance loads for channels. However, this scheme does not guarantee any source-destination pair has the minimum hops, while our goal is to construct the interconnection networks with small diameters. Besides, locality-preserving randomized oblivious(RLB) [48] algorithm, globally oblivious adaptive locally algorithm (GOAL) [46], and globally adaptive load-balanced (GAL) routing [47], etc., have also been proposed to optimize the load during transmission. But the deterministic routing is common in practice because it is easy to deploy and implement, meanwhile, the adaptive routing should introduce out-of-order packets, that would cost more channel resources [23]. Gomez presented an amazing result that deterministic routing could achieve similar performance in some scenarios than adaptive routing for fat-tree topology [30]. However, there's very little research on deterministic routing for regular networks. Therefore, we proposed a novel optimal deterministic routing based on mixed-integer nonlinear programming and be solved strictly to implement load balancing. In addition to its application to a single network, we also find this method can be applied to product networks easily, which could make us implementing balanced routing for large scale networks.

For evaluating the efficiency of optimal routing, we recalibrated the Graph500 [11] benchmark on a Beowulf [18] cluster. For the previous research of enhanced network by using the optimal graph, we focused on the parameter of topology and only considered the shortest length path in routing, but ignored the balancing for routing. Hence, we selected the Graph500, stressed all to all communication performance, to evaluate the effect of the optimal routing algorithm, which shows sustainable improvement of throughput. For the

Table 1: Properties of classical optimal graphs

Graph	Vertices	Degree	Diameter	MPL	BW
Peterson[22]	10	3	2	1.67	5
Heawood[8]	14	3	3	2.07	7
Levi[42]	30	3	4	2.86	9
(16,3)-optimal[14]	16	3	3	2.20	6
(16,4)-optimal[14]	16	4	3	1.75	12
(32,3)-optimal[14]	32	3	4	2.94	10
(32,4)-optimal[14]	32	4	3	2.94	16

larger network, we used the SimGrid, a framework to evaluate and compare relevant platform configurations, algorithmic approaches, and system designs, to benchmark for our novel routing algorithm due to hardware limitations. It is amazing that the optimal routings improve the performance drastically for some applications depend on all-to-all communication. In summary, combing the optimal routing with MDBs, that can provide a series of scalable networks with high throughput for the next-generation supercomputers.

The rest of the paper is organized as follows; section 2 introduces the optimal regular graph and discusses the topology architecture of MDBs. The vertex-balanced routing is also involved in section 2. Section 3 shows benchmark results from different scales of networks, which come from a Beowulf cluster and simulation framework. Section 4 concludes the paper.

2 Topology and routing

2.1 The optimal regular graph

The optimal regular graph is defined to be the N-vertex degree-k regular graph with minimal mean path length (MPL) and can be denoted by (N, k). If $k = 2$, the network degenerates to a ring network. Petersen graph a classic example with $N = 10, k = 3$. Cerf et al. [10] proved that the diameter of any regular graph with minimal MPL is also minimal and got the lower bound MPL for (N, k).

Usually, the brute-force search for optimal graph (N, k) with small N is effective. Yuefan[14] et al. reported graphs with minimal diameter for $(16, 3)$,$(16, 4)$,$(32, 3)$ and $(32, 4)$ by using parallel exhaustive search method [55] and presented the excellent benchmark results for a series of optimal regular networks. Figure 1 shows the optimal graphs for 16-vertex and 32-vertex from Yuefan[14] et al, while Table 1 shows the vertices, degree, diameters, MPL and bisection width(BW) of them and some classical low-radix graphs.

However, when N becomes larger, the computation of exhausted is expensive and can't be accepted. Thus some heuristic methods have been proposed to solve this problem. Additionally, its similar problem named as degree/diameter

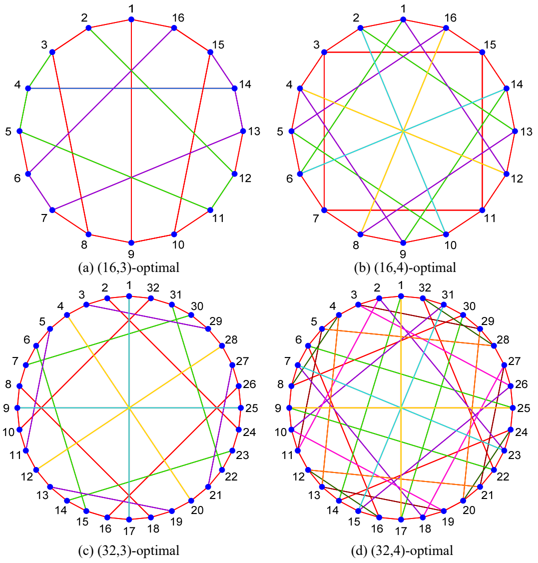


Fig. 1: The optimal graphs used by a group [14] constructing a cluster.

problem also attracted much attention. Mizuno [36] et.al. reported the result for $(256, 22)$. Obviously, exhausted and heuristic methods still cost much computation resources. Therefore, we proposed the minimum-diameter-based scalable networks(MDBs) to construct large scale regular graphs from small scale graphs with minimum diameter by using the Cartesian product.

2.2 Product networks

The Cartesian product method is a fast and valuable method to construct a larger scale interconnection network by using specified [53].

Definition 1 The Cartesian product $G = G_1 \otimes G_2$ of two undirected connected graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ is also the undirected graph $G = (V, E)$, where the vertex for G can be presented as a two-tuples $\langle u_1, u_2\rangle$ and $V = \{\langle u_1, u_2\rangle | u_1 \in V_1$ and $u_2 \in V_2\}$, and the set of edges is $E = \{(\langle u_1, u_2\rangle, \langle v_1, v_2\rangle) | ((u_1 = v_1)$ and $(u_2, v_2) \in E_2)$ or $((u_2 = v_2)$ and $(u_1, v_1) \in E_1)\}$.

Figure. 2 shows an example of the Cartesian product for $(8, 3)$ and 4-vertex complete graph. We can see that all edges for $\langle *, 1\rangle$ labeled as bold red line are organized with G_1, then repeated as blue lines. In addition, all of $\langle a, *\rangle$ are all connected same with G_2. If $G_1 = G_2$, the graph G is named as folded graph, and G_1 is Peterson graph, we can get folded Peterson Graph(FP_n) [39].

The Cartesian product have many desirable properties such as

- the size $S()$ of $G_1 \otimes G_2$ is $S(G) = S(G_1) \cdot S(G_2)$,

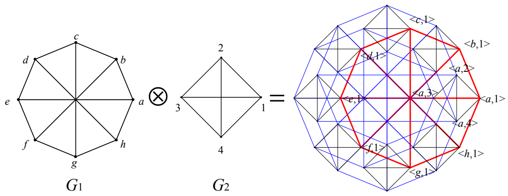


Fig. 2: An example of $(8, 3) \otimes (4, 3)$.

Table 2: Properties of folded networks

Topology	# Nodes	Diameter	Degree
Hypercube	2^n	n	n
Folded Peterson[40]	10^n	$2n$	$3n$
Folded Heawood[24]	14^n	$3n$	$3n$
Folded Levi	30^n	$4n$	$3n$
Folded-(16,3)	16^n	$3n$	$3n$
Folded-(16,4)	16^n	$3n$	$4n$
Folded-(32,3)	32^n	$4n$	$3n$
Folded-(32,4)	32^n	$3n$	$4n$

- the degree $d()$ of $G_1 \otimes G_2$ is $d(G) = d(G_1) + d(G_2)$,
- the diameter $\delta()$ of $G_1 \otimes G_2$ is $\delta(G) = \delta(G_1) + \delta(G_2)$,
- $G_1 \otimes G_2$ is isomorphic to $G_2 \otimes G_1$

These properties are easy to prove[13, 5] and they imply that the diameter grow linearly while the scale of the network increases exponentially, which is the desired feature to design large scale interconnection network.

Using the Cartesian product, with optimal graphs [14] that discovered and verified recently, we can build scalable networks. Fig. 3 and Tab. 2 show diameters of folded networks, we can see that hypercube can provide the most diversities of scales, but the diameters of folded Peterson networks are minimal, same to folded-(32,4), while other networks balanced the diameters and diversities. Here, the diameter of folded-(16,4) is same as folded-(16,3), but the more degrees of nodes would reduce the forwarding loads and improve the throughput of networks; from the aspect of routing, the low degree is not a good choice, whose affection on the network performance may eliminate the saving of hardware ports and wires.

For small-scale networks, the MDBs can also prove many choices as shown in Fig. 4, when the number of nodes is less than 1,000, the folded Petersen network marked as triangle only support scales of 10, 100 and 1,000, while the hypercube networks have the most scale diversities, though the diameters of hypercube networks as not the optimal. The combinations of $(8, 3) \otimes (8, 3)$, $(32, 4) \otimes (8, 3)$, and $(32, 4) \otimes (32, 4)$, etc. can fill the gaps of folded Peterson networks, which also reduce latencies of networks because of low-radix.

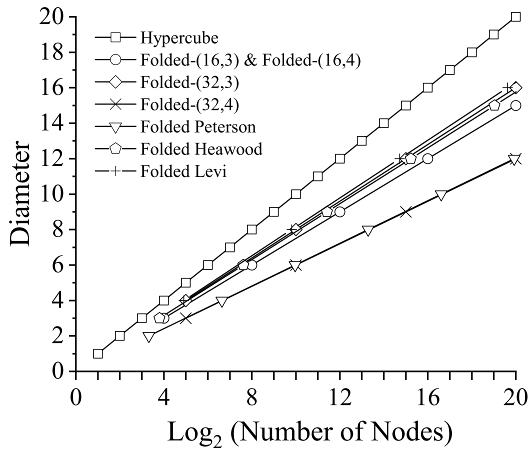


Fig. 3: The diameters of different folded networks.

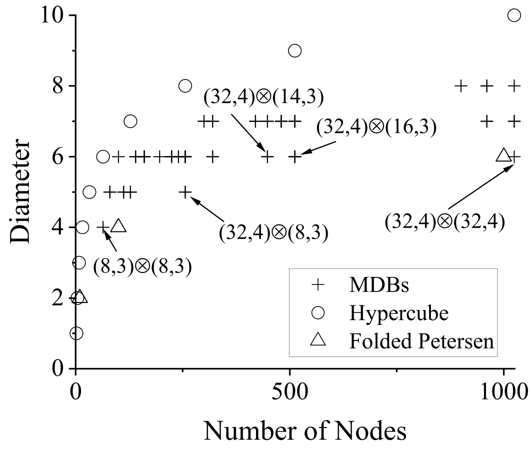


Fig. 4: The diameters of the MDBs that contain multi-type base graphs.

2.3 Vertex-balanced routing algorithm

In direct networks, each node exchange messages via other nodes in most conditions, except they are wired directly, where each node is not going to do computing but also need to forward data from other nodes. In this condition, the imbalanced load will lead to blocking and slow down the computing speed of nodes, because interrupts caused by packets would occupy the processor resource. With consideration to balance the forwarding pressure, we proposed the vertex-balanced algorithm to balance the load, while keep the shortest hops between each node.

The routing algorithm can be described in two stages. The first stage of this algorithm is to enumerate all the shortest paths between any two nodes. Then we will choose the path in the next stage according to the solutions of programming if there existed two or more shortest paths between two nodes.

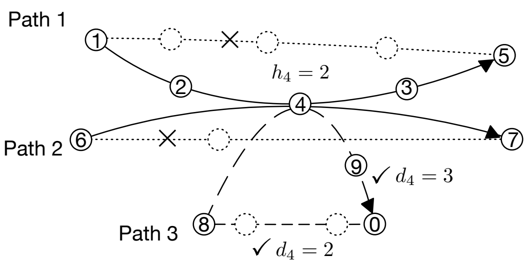
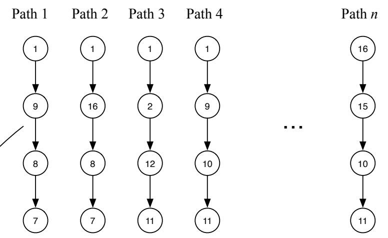


Fig. 5: The diagram of variables d_k and h_k. The dotted lines with cross sign mean that the source-destination pair only have one shortest path, and the dashed lines mean that there exist multi-path between node pair.

Here, we use d_k as showing in Fig. 5 to measure the load of node k, which represents the total number of times when the paths get through node k. For example, we define the fixed load of node 4 $h_4 = 2$ in Fig.2, because there only exists one possible the shortest path for node pairs 1-5, 6-7, represented by dotted lines, and node 4 is located on these paths. But the shortest paths of 3 hops, between node pair 8-0, represented by dashed lines, have two candidates; the node labeled as 4 is involved in one path, but not in the other one. Hence, the load for node 4 is $d_4 = 3$, if the upper path has been selected. For balancing the load for each node, the second stage can be written in mixed-integer nonlinear programming (MINLP) as follows,

$$\begin{aligned} \min \quad & \sum_{k=1}^{N} \left(d_k - \frac{\sum d_k}{N} \right)^2 \\ \text{Subject to:} \quad & \forall s_i \in \{0,1\}, \forall p_{kj} \in \{0,1\} \\ & \sum_{i=1}^{n_{\text{group}}} g_{ij} s_i = 1 \\ & d_k = h_k + \sum_{j=1}^{n_{\text{route}}} p_{kj} s_i \end{aligned} \tag{1}$$

In this problem, N is vertices and p_{kj} is the matrix indicated whether node k located in path j when there exist multi-path for source-destination pair, but the start point and endpoint are not included. As shown in Fig. 6, there are two choices, $1 \rightarrow 9 \rightarrow 8 \rightarrow 7$ and $1 \rightarrow 16 \rightarrow 8 \rightarrow 7$, for node pair 1-7. Therefore, corresponding positions of the matrix p_{kj} would be set as 1. In this matrix, the labels of paths are arbitrary because we will introduce the group matrix g_{ij} to distinguish them.



Node	Path 1	Path 2	Path 3	Path 4	Path n
1	0	0	0	0	0
2	0	0	1	0	0
3	0	0	0	0	0
4	0	0	0	0	0
5	0	0	0	0	0
6	0	0	0	0	0
7	0	0	0	0	0
8	1	1	0	0	0
9	1	0	0	1	0
10	0	0	0	1	1
11	0	0	0	0	0
12	0	0	1	0	0
13	0	0	0	0	0
14	0	0	0	0	0
15	0	0	0	0	1
16	0	1	0	0	0

p_{kj}

Fig. 6: The diagram of variable p_{kj}.

The matrix g_{ij} is the group matrix, which puts the same source-destination pairs into one group, as shown in Fig. 7. For example, the source-destination pair of path 1 and path 2 is $1 \to 7$, so $g_{11} = g_{12} = 1$ when we label these two paths as Group1. In addition to consideration of the group matrix, the constraint of $\sum s_j = 1$ hint that one source-destination pair only has one path, for example, we can see $s_1 + s_2 = 1$ or $s_3 + s_4 = 1$ in Fig.3. It should be noticed that we did not define the direction in this algorithm, but directed routing would be effective than undirected routing in practice because directed routing can provide more choices to avoid congestion than undirected routing. The objective function is nonlinear, but this MINLP problem can be solved by using some packages such as LINGO [38], IBM ILOG Cplex [7], MOSEK [6], and so on. Here, we use LINGO with an educational research license to solve this programming. In addition to commercial or open-source packages, heuristic methods such as genetic algorithm and simulated annealing can be used to approaches this problem approximately.

Proposition 1 *For the product networks, if the condition of* $\sum_{k=1}^{N} (d_k - \sum d_k/N)^2 = 0$ $(d_1 = d_2 = d_3 \ldots)$ *for* G_1, G_2 *is satisfied, for* $G_1 \times G_2$, *this condition is also satisfied.*

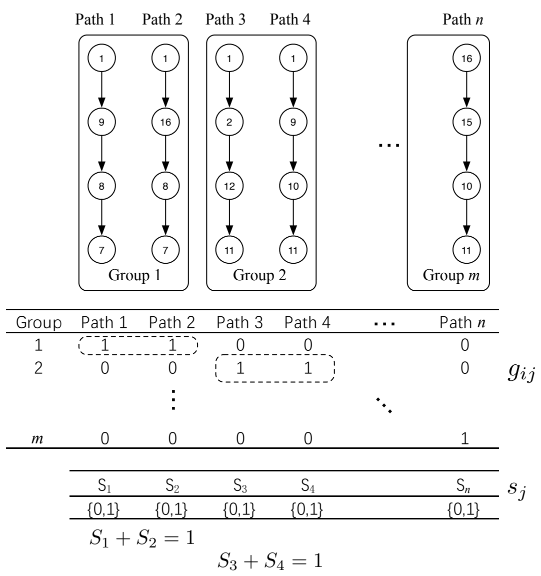


Fig. 7: The diagram of variables g_{ij} and s_j.

We can prove this proposition from different type of source-destination pairs. Now we define $u \in G_1$ and $v \in G_2$, there existed a path from $\langle u_1, v_1\rangle \rightarrow \langle u_2, v_2\rangle$, we can discuss as follows,

1. $u_1 = u_2$
   That means the path is located in G_2, it is obviously that $d_{\langle u_1,v_1\rangle} = d_{\langle u_1,v_2\rangle} = \cdots$
2. $v_1 = v_2$
   Similarly, this path is located in G_1, then $d_{\langle u_1,v_1\rangle} = d_{\langle u_2,v_1\rangle} = \cdots$
3. $u_1 \neq u_2$ and $v_1 \neq v_2$
   In this condition, the path can be expressed as two paths in the G_1 and G_2 independently.

$$\begin{aligned} &\langle u_1, v_1\rangle \rightarrow \langle u_2, v_2\rangle \\ \Rightarrow &\langle u_1, v_1\rangle \rightarrow \langle u_2, v_1\rangle \rightarrow \langle u_2, v_2\rangle \\ \text{or } \Rightarrow &\langle u_1, v_1\rangle \rightarrow \langle u_1, v_2\rangle \rightarrow \langle u_2, v_2\rangle \end{aligned} \tag{2}$$

   Here, the routes of $\langle u_1, v_1\rangle \rightarrow \langle u_2, v_1\rangle$ and $\langle u_2, v_1\rangle \rightarrow \langle u_2, v_2\rangle$ have also been proved above. Hence, the conclusion is also be satisfied in this condition, which also provide the routing method for product networks.

Additionally, the conclusion is not to be satisfied when $\sum(d_k - \sum d_k/N)^2 \neq 0$ but the load of $G_1 \times G_2$ is near to balance status. When N is larger, the size of matric increases rapidly, which limits us to solve this problem directly. Therefore, this feature is significant because we can obtain the optimal routing of larger networks from small-scale networks by using product networks, which also provide a new method to solve the special case of problem 1.

3 Benchmark

To verify the assumption that balanced routing would enhance the network, we performed some experiments, including benchmark on a real cluster and simulations by using SimGrid [9]. For small-scale networks and base graphs, we used a Beowulf cluster in Jinan[14] to recalibrate some results. Unluckily, the simulation is the feasible choice for benchmarking the larger network, because it is hard to find a prototype machine with hundreds of nodes for us to change the wiring.

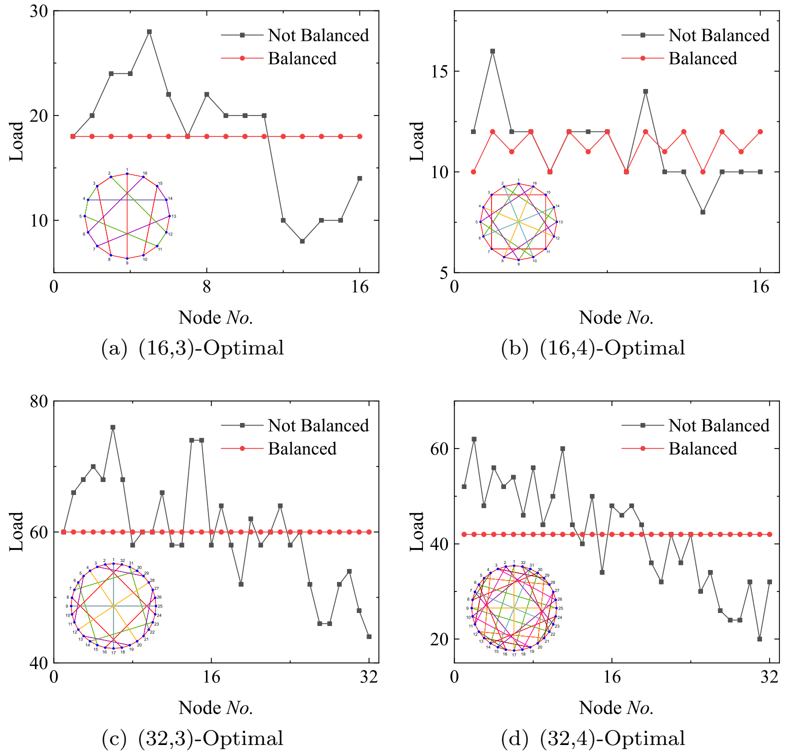
3.1 Graph500 on Taishan

To evaluate the performance of balance routing, we perform the experiments on a Beowulf cluster with hardwares designed for soft routers, and we name it as Taishan, which is a functional prototype platform for optimal topology research. The cluster contains 36 nodes and detailed hardware configuration is shown as follow,

- Processor: Intel Celeron 1037U
- Memory: 1×8 GB General DDR3 SODIMM (1600 MHz, 1.35V)
- Internal Storage: 128 GB General SSD
- Network Storage: NFS via 48-port Gigabit Ethernet switch
- Ethernet: Intel 82583V Gigabit Ethernet controllers (8 ports)
- Operation System: CentOS Linux 6.7 (kernel 2.6.32)
- Compiler: GCC 4.4.7
- MPI Environment: MPICH 3.2

The configuration is designed to adapt to network switching, but not for high-density computing. However, this is the economical option after carefully chosen, because the spot we focused on is topology and higher performance would need more budget. During previous research of enhanced topology, we adopt static routing only according to Floyd algorithm to ensure the shortest path length but not consider the load balance.

Fig. 8 compares the difference of loading of each topology between not balanced routing for previous research and load-balance routing, the load for each node fluctuates in an extensive range, which leads to congestion in some nodes and reduces the efficiency of the whole network. Except for the topology of $(16, 4)$, the objective function of optimal routing can converge to zero, that means numbers of paths through every node is balanced, to prevent the situation some flows focus on one node. For the topology of $(16, 4)$ as shown in Fig. 1 (b), the range of load is located in the range of $(10, 12)$, which is better than original routing, although the objective function does not converge to zero.



(a) (16,3)-Optimal

(b) (16,4)-Optimal

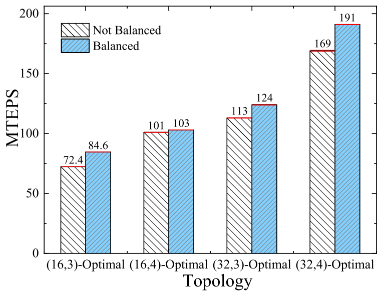
(c) (32,3)-Optimal

(d) (32,4)-Optimal

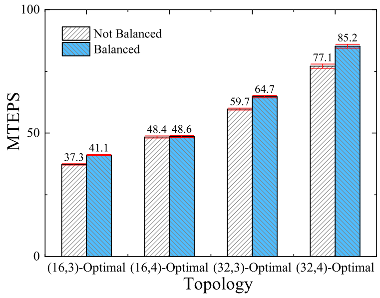
Fig. 8: Forwarding loads of each topology.

According to the analysis above, the new routing strategy optimizes the throughput for global communication, hence we select the Graph 500 (version 3.0.0), a benchmark baseline of multiple breadth-first searches (BFS) and single-source shortest path (SSSP) computations, to exhaust extremely large undirected graph on distributed networks, which help stress communication subsystem and data-intensive performance of multi-computer clusters. Graph 500 uses mean TEPS (traversed edges per second) as the metric to evaluate the performance. Here, the test scale was set as 27, which is the same as previous research and would generate an initial unweighted graph for BFS with the size of 24 GB, correspondingly, the size of the initial weighted graph for SSSP reach to 40 GB, lead to massive message exchanging that can help stress the global communication capacity.

By using the Graph 500 benchmark with balanced routing, we recalibrate performance data of previous results. Fig. 9 shows the latest results, and the red lines represent standard deviation, which compensates for the weaknesses for previous results obviously, the speed of improved, especially for the topology with order 23, by adapting balanced routing. Among these results, the effect for $(16, 4)$ is less than other topologies, because the imbalances for them



(a) Graph500-BFS



(b) Graph500-SSSP

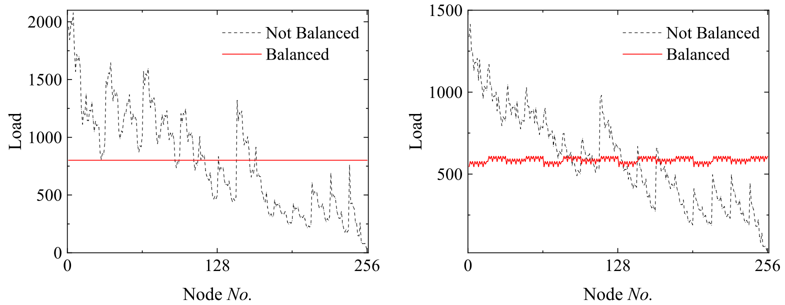
Fig. 9: Recalibration results of Graph 500 for [14].

are severe than $(16,4)$, while the objective function of balanced routing for $(16,4)$ can not converge to 0 strictly as shown in Fig.1(b). Generally, the balanced routing strategy can improve the performance of some applications that depend on massive communications.

3.2 Simulation

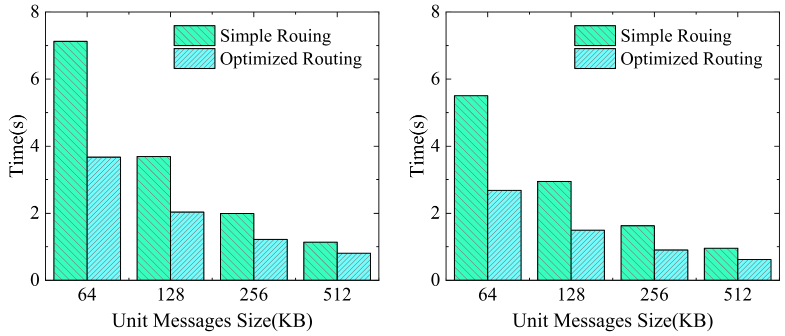
The evaluation of larger-scale topologies generated from product networks performs on the platform SimGrid, similar to NS [16], which concentrates on the aspect of the network, provides the accurate, versatile, and scalable simulation for distributed computing or cloud platform, especially with SMPI, help simulate MPI application with no or less modification [9]. We use these parameters: 8 Gflops processing speed per core, full-duplex gigabit link and 30 μs latency per link, which approximate the ping-pong test results of the "Taishan" cluster, but different with the configurations of "Taishan," we adopt the model of one-core CPU per host, to eliminate the affection of computer architecture. We use two static routing tables, one is generated from Floyd algorithm without considering balanced routing, and the other one comes from combing the optimal routing of the small-scale graph and product networks, to do comparative simulations on SimGrid. All of the simulations run on the SeaWulf cluster at Stony Brook University.

Fig. 10 shows the load with not balanced routing and balanced routing for two product networks of $(16,3) \otimes (16,3)$ and $(16,4) \otimes (16,4)$. In Fig. 10 (a), each node has the same load, due to the solution for $(16,3)$ can converge to 0 as shown in Fig. 8 (a), but strategy generated from the Floyd algorithm leads to an extensive fluctuation of loads, especially, the max load is about 2000, so many paths passing through one node would cause congestion, while this case verified Proposition 1. In Fig. 10 (b), the balanced routing still has a little fluctuation, because the objective function of balanced routing for (16,4) can't converge to 0 strictly, which is shown in Fig. 8 (b); however, the balanced routing still reduces the difference of load for each node.



(a) $(16,3)\otimes(16,3)$ (b) $(16,4)\otimes(16,4)$

Fig. 10: Forwarding loads of $(16,3)\otimes(16,3)$ and $(16,4)\otimes(16,4)$.

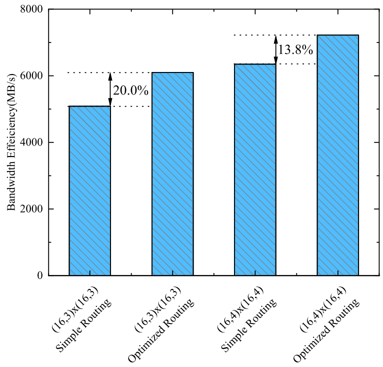


(a) $(16,3)\otimes(16,3)$ (b) $(16,4)\otimes(16,4)$

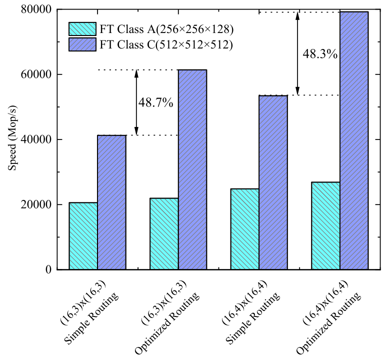
Fig. 11: Benchmark results for MPI All-to-all.

With balanced routing, we use MPI All-to-all, effective bandwidth, and NPB FT, benchmarks largely depend on global communication, to survey the effect of balanced routing algorithm, but except for Graph 500 (version 3.0.0), due to lack of function of MPI_Ibarrier in SimGrid. Additionally, almost all network simulators including SimGrid and NS only support a single core, with the 128 GB RAM for computing nodes on SeaWulf, that limit the scale of topology and complexity of applications, especially for high-density computing. Hence, the product networks of $(16,4)\otimes(16,4)$ and $(16,3)\otimes(16,3)$ are be used to evaluate the performance the balanced routing, while unit message sizes for MPI All-to-All are not over 512KB, maximum message size for effective bandwidth benchmark is 1MB, and the NPB FT (3D-FFT benchmark) use two types of Class A, whose size is $256\times256\times128$, and Class C, whose size is $512\times512\times512$.

Fig. 11 presents the results of MPI All-to-all, the time decrease obviously, even half than the routings without optimization, especially for the small packet, because larger packets stress the bandwidth than small pack-



(a) Bandwidth efficiency



(b) NPB FT

Fig. 12: Benchmark results of bandwidth efficiency and NPB FT.

ets. Fig. 12 (a) is the comparison for bandwidth efficiency, total bandwidths of the network have been increased by 20.0% of $(16,3) \otimes (16,3)$, and 13.8% of $(16,4) \otimes (16,4)$. For the benchmark of NPB FT, performances increase tremendously, especially for Class C contains more data. All benchmark results and the forwarding loads between $(16,3) \otimes (16,3)$ and $(16,4) \otimes (16,4)$ show that high degrees would reduce the average forwarding load and improve the throughput of the network, though more ports and links would lead more cost.

In summary, whether experiments on a Beowulf machine or simulations on SimGrid, no matter for small scale topology or larger product networks, the results show that the balanced routing enhanced performances of networks. The method can be expanded for multi-level product networks, to construct the larger networks for the next-generation supercomputer.

4 Conclusions

In this paper, we propose the product networks contain a diversity of base graphs, not just the Peterson graph, but the optimal graphs of any order, which can create a scalable, modularity topologies fit for the high-performance computing. In addition to its scalable, we also present a method to optimize the routing, which can implement the load balance for static routing. For the product networks, it is easy to prove that this method is easy to expand and still keep load balance for each node, which improves the throughput of networks with MDBs. To verify the effect of balanced routing, recalibration of Graph 500 for our previous work on the enhanced networks and simulations for product networks show that balanced routing can also enhance the networks. The MDBs with optimal routing can be a potential candidate for building the cluster of different scales in the future.

Acknowledgments

Z. Xu is supported by the special project high performance computing of National Key Research and Development Program 2016YFB0200604. We thank the National Supercomputing Centers in Jinan, China, for support of constructing a Beowulf cluster Taishan, the Computing Center (SeaWulf) at Stony Brook University, for support of simulation resources. We also thank the following individuals for their contributions to this work at different stages, M. Guo, W. Liu, L. Pei, and W. Cun et al.

References

1. Top 500 Supercomputer Site (2018). http://www.top500.org
2. Abd-El-Barr, M., Al-Somani, T.F.: Performance comparison of hierarchical interconnection networks. In: Proceedings of 2011 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing. IEEE (2011). DOI 10.1109/pacrim.2011.6032891
3. Adiga, N.R., Blumrich, M.A., Chen, D., Coteus, P., Gara, A., Giampapa, M.E., Heidelberger, P., Singh, S., Steinmacher-Burow, B.D., Takken, T., Tsao, M., Vranas, P.: Blue Gene/L torus interconnection network. IBM Journal of Research and Development **49**(2.3), 265–276 (2005). DOI 10.1147/rd.492.0265
4. Ajima, Y., Sumimoto, S., Shimizu, T.: Tofu: A 6d mesh/torus interconnect for exascale computers. Computer **42**(11), 36–40 (2009). DOI 10.1109/mc.2009.370
5. Al-Ayyoub, A.E., Day, K.: Comparative study of product networks. Journal of Parallel and Distributed Computing **62**(1), 1–18 (2002). DOI 10.1006/jpdc.2001.1770
6. Andersen, E.D., Andersen, K.D.: The mosek interior point optimizer for linear programming: An implementation of the homogeneous algorithm. In: Applied Optimization, pp. 197–232. Springer US (2000)
7. Bliek1ú, C., Bonami, P., Lodi, A.: Solving mixed-integer quadratic programming problems with ibm-cplex: a progress report. In: Proceedings of the twenty-sixth RAMP symposium, pp. 16–17 (2014)
8. Brinkmann, G., Coolsaet, K., Goedgebeur, J., Mélot, H.: House of graphs: A database of interesting graphs. Discrete Applied Mathematics **161**(1-2), 311–314 (2013). DOI 10.1016/j.dam.2012.07.018
9. Casanova, H., Giersch, A., Legrand, A., Quinson, M., Suter, F.: Versatile, scalable, and accurate simulation of distributed applications and platforms. Journal of Parallel and Distributed Computing **74**(10), 2899–2917 (2014). DOI 10.1016/j.jpdc.2014.06.008
10. Cerf, V.G., Cowan, D.D., Mullin, R.C., Stanton, R.G.: A lower bound on the average shortest path length in regular graphs. Networks **4**(4), 335–342 (1974). DOI 10.1002/net.3230040405
11. Checconi, F., Petrini, F.: Massive data analytics: The graph 500 on IBM blue gene/q. IBM Journal of Research and Development **57**(1/2), 10:1–10:11 (2013). DOI 10.1147/jrd.2012.2232414
12. Das, S., Banerjee, A.: Hyper petersen network: yet another hypercube-like topology. In: [Proceedings 1992] The Fourth Symposium on the Frontiers of Massively Parallel Computation. IEEE Comput. Soc. Press (1992). DOI 10.1109/fmpc.1992.234949
13. Day, K., Al-Ayyoub, A.E.: The cross product of interconnection networks. IEEE Transactions on Parallel and Distributed Systems **8**(2), 109–118 (1997). DOI 10.1109/71.577251
14. Deng, Y., Guo, M., Ramos, A.F., Huang, X., Xu, Z., Liu, W.: Optimal low-latency network topologies for cluster performance enhancement. arxiv (2019)
15. Efe, K.: A variation on the hypercube with lower diameter. IEEE Transactions on Computers **40**(11), 1312–1316 (1991). DOI 10.1109/12.102840

16. Fall, K.: Network emulation in the VINT/NS simulator. In: Proceedings IEEE International Symposium on Computers and Communications (Cat. No.PR00250). IEEE Comput. Soc. DOI 10.1109/iscc.1999.780820
17. Fu, B., Xu, S., Bao, W., Jiang, G., Chen, M., Zhang, L., Tao, Y., He, R., Zhao, J.: Dandelion: A locally-high-performance and globally-high-scalability hierarchical data center network. In: 2014 23rd International Conference on Computer Communication and Networks (ICCCN). IEEE (2014). DOI 10.1109/icccn.2014.6911750
18. Gropp, W., Lusk, E.L., Sterling, T. (eds.): Beowulf Cluster Computing with Linux (Scientific and Engineering Computation). The MIT Press (2003)
19. Hafner, P.R.: Journal of Algebraic Combinatorics **18**(1), 7–12 (2003). DOI 10.1023/a:1025136524481
20. Haq, E.: Cross-cube: a new fault tolerant hypercube-based network. In: [1991] Proceedings. The Fifth International Parallel Processing Symposium. IEEE Comput. Soc. Press. DOI 10.1109/ipps.1991.153821
21. Harary, F., Hayes, J.P., Wu, H.J.: A survey of the theory of hypercube graphs. Computers & Mathematics with Applications **15**(4), 277–289 (1988). DOI 10.1016/0898-1221(88)90213-1
22. Holton, D.A., Sheehan, J.: The Petersen Graph, vol. 7. Cambridge University Press (1993)
23. James, W., Patrick, B.: Principles and Practices of Interconnection Networks. Elsevier Science & Technology (2003)
24. Jan, G.E., Hwang, Y., Lin, M., Liang, D.: Novel hierarchical interconnection networks for high-performance multicomputer systems. Journal of information science and engineering **20**, 1213–1229 (2004)
25. Kim, J., Dally, W.J., Scott, S., Abts, D.: Technology-driven, highly-scalable dragonfly topology. In: 2008 International Symposium on Computer Architecture. IEEE (2008). DOI 10.1109/isca.2008.19
26. Kini, N.G., Kumar, M.S., Mruthyunjaya, H.: Torus embedded hypercube interconnection network: A comparative study. International Journal of Computer Applications **1**(4), 32–35 (2010). DOI 10.5120/106-217
27. Kitasuka, T., Iida, M.: A heuristic method of generating diameter 3 graphs for order/degree problem (invited paper). In: 2016 Tenth IEEE/ACM International Symposium on Networks-on-Chip (NOCS). IEEE (2016). DOI 10.1109/nocs.2016.7579334
28. Koibuchi, M., Fujiwara, I., Fujita, S., Nakano, K., T. Uno, T.I., Kawarabayashi, K.: Graph Golf: The Order/degree Problem Competition. `http://research.nii.ac.jp/graphgolf/`
29. Kumar, J., Patnaik, L.: Extended hypercube: a hierarchical interconnection network of hypercubes. IEEE Transactions on Parallel and Distributed Systems **3**(1), 45–57 (1992). DOI 10.1109/71.113081
30. Leiserson, C.E.: Fat-trees: Universal networks for hardware-efficient supercomputing. IEEE Transactions on Computers **C-34**(10), 892–901 (1985). DOI 10.1109/tc.1985.6312192
31. Liao, X.K., Pang, Z.B., Wang, K.F., Lu, Y.T., Xie, M., Xia, J., Dong, D.Z., Suo, G.: High performance interconnect network for tianhe system. Journal of Computer Science and Technology **30**(2), 259–272 (2015). DOI 10.1007/s11390-015-1520-7
32. Louri, A., Sung, H.: Scalable optical hypercube-based interconnection network for massively parallel computing. Applied Optics **33**(32), 7588 (1994). DOI 10.1364/ao.33.007588
33. Louri, A., Weech, B., Neocleous, C.: A spanning multichannel linked hypercube: a gradually scalable optical interconnection network for massively parallel computing. IEEE Transactions on Parallel and Distributed Systems **9**(5), 497–512 (1998). DOI 10.1109/71.679219
34. Malluhi, Q., Bayoumi, M.: The hierarchical hypercube: a new interconnection topology for massively parallel systems. IEEE Transactions on Parallel and Distributed Systems **5**(1), 17–30 (1994). DOI 10.1109/71.262585
35. Miller, M., Sirán, J.: Moore graphs and beyond: A survey of the degree/diameter problem. The electronic journal of combinatorics **1000**, DS14–Dec (2005)

36. Mizuno, R., Ishida, Y.: Constructing large-scale low-latency network from small optimal networks. In: 2016 Tenth IEEE/ACM International Symposium on Networks-on-Chip (NOCS). IEEE (2016). DOI 10.1109/nocs.2016.7579336
37. Moore, G.: Cramming more components onto integrated circuits. Proceedings of the IEEE **86**(1), 82–85 (1998). DOI 10.1109/jproc.1998.658762
38. Neumaier, A., Shcherbina, O., Huyer, W., Vinkó, T.: A comparison of complete global optimization solvers. Mathematical Programming **103**(2), 335–356 (2005). DOI 10.1007/s10107-005-0585-4
39. Ohring, S., Das, S.: Folded petersen cube networks: new competitors for the hypercubes. In: Proceedings of 1993 5th IEEE Symposium on Parallel and Distributed Processing. IEEE Comput. Soc. Press (1993). DOI 10.1109/spdp.1993.395482
40. Ohring, S., Das, S.K.: The folded petersen network : A new communication-efficient multiprocessor topology. In: 1993 International Conference on Parallel Processing - ICPP 93 Vol1. IEEE (1993). DOI 10.1109/icpp.1993.175
41. Pan, Y., Chuang, H.: Properties and performance of the block shift network. IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications **44**(2), 93–102 (1997). DOI 10.1109/81.554320
42. Randi, M.: Symmetry properties of graphs of interest in chemistry. II. desargues-levi graph. International Journal of Quantum Chemistry **15**(6), 663–682 (1979). DOI 10.1002/qua.560150610
43. Seo, J.H., Kim, J.S., Chang, H.J., Lee, H.O.: The hierarchical petersen network: a new interconnection network with fixed degree. The Journal of Supercomputing **74**(4), 1636–1654 (2017). DOI 10.1007/s11227-017-2186-4
44. Shi, W., Srimani, P.K.: Hierarchical star: a new two level interconnection network. Journal of Systems Architecture **51**(1), 1–14 (2005). DOI 10.1016/j.sysarc.2004.05.003
45. Shi, Y., Hou, Z., Song, J.: Hierarchical interconnection networks with folded hypercubes as basic clusters. In: Proceedings Fourth International Conference/Exhibition on High Performance Computing in the Asia-Pacific Region. IEEE (2000). DOI 10.1109/hpc.2000.846533
46. Singh, A., Dally, W., Gupta, A., Towles, B.: GOAL: a load-balanced adaptive routing algorithm for torus networks. In: 30th Annual International Symposium on Computer Architecture, 2003. Proceedings. IEEE Comput. Soc (2003). DOI 10.1109/isca.2003.1207000
47. Singh, A., Dally, W., Towles, B., Gupta, A.: Globally adaptive load-balanced routing on tori. IEEE Computer Architecture Letters **3**(1), 2–2 (2004). DOI 10.1109/l-ca.2004.8
48. Singh, A., Dally, W.J., Towles, B., Gupta, A.K.: Locality-preserving randomized oblivious routing on torus networks. In: Proceedings of the fourteenth annual ACM symposium on Parallel algorithms and architectures - SPAA '02. ACM Press (2002). DOI 10.1145/564870.564873
49. Teh, M.Y., Wilke, J.J., Bergman, K., Rumley, S.: Design space exploration of the dragonfly topology. In: Lecture Notes in Computer Science, pp. 57–74. Springer International Publishing (2017)
50. Valiant, L.G., Brebner, G.J.: Universal schemes for parallel communication. In: Proceedings of the thirteenth annual ACM symposium on Theory of computing - STOC '81. ACM Press (1981). DOI 10.1145/800076.802479
51. Wang, H., Xu, D., Li, L.: A novel globally adaptive load-balanced routing algorithm for torus interconnection networks. ETRI Journal **29**(3), 405–407 (2007). DOI 10.4218/etrij.07.0206.0241
52. Wu, C.L., Feng, T.Y.: On a class of multistage interconnection networks. IEEE Transactions on Computers **C-29**(8), 694–702 (1980). DOI 10.1109/tc.1980.1675651
53. Xu, J.: Topological Structure and Analysis of Interconnection Networks. Springer US (2002)
54. Xu, M., Xu, J.M., Hou, X.M.: Fault diameter of cartesian product graphs. Information Processing Letters **93**(5), 245–248 (2005). DOI 10.1016/j.ipl.2004.11.005
55. Xu, Z., Huang, X., Jimenez, F., Deng, Y.: A new record of enumeration of regular graphs by parallel processing (2019)
56. Youssef, A.: Design and analysis of product networks. In: Proceedings Frontiers '95. The Fifth Symposium on the Frontiers of Massively Parallel Computation. IEEE Comput. Soc. Press (1995). DOI 10.1109/fmpc.1995.380472