JOURNAL OF
Optical Communications
and Networking

Hierarchical and reconfigurable optical/electrical interconnection network for high-performance computing

ZUOQING ZHAO, BINGLI GUO, YU SHANG, AND SHANGUO HUANG*

School of Optoelectronic Information (Institute of Information Photonics and Optical Communications), Beijing University of Posts and Telecommunications, Beijing 100876, China
**Corresponding author: shghuang@bupt.edu.cn*

Received 10 September 2019; revised 11 December 2019; accepted 16 December 2019; published 30 January 2020 (Doc. ID 377427)

Compared with electrical packet switches, optical switching technology could enable a more desirable high-performance computing (HPC) system with lower power consumption, lower delay, higher bandwidth, and more flexibility. In this article, we designed a hierarchical and reconfigurable optical/electrical HPC interconnection network. The traffic matrix of the target task can be decomposed into multiple matrix groups that are executed in parallel, and the network topology in each layer can be reconfigured according to the subtraffic matrix associated with this layer. For interlayer cross-connection, we use a shuffle network to offer a direct optical bypass for a huge aggregated amount of interlayer communication requirements, as well as multiple light paths. We propose a reconfiguration optimization algorithm and routing algorithm to optimize network performance. Simulation results show that, with the proposed architecture and algorithms, the average path length per unit of communication intensity is minimized by 13.7%–52.4%, the delay is reduced by more than 14.8%, and throughput is improved by 33% at least compared with Mesh, Torus, and Dragonfly.

https://doi.org/10.1364/JOCN.377427

1. INTRODUCTION

With the increasing demand for processing capability, high-performance computing (HPC) is now accepted as a key enabling technology for scientific and industrial application. Researchers are continuously working on this topic to improve peak performance of HPC systems. To sustain this trend, data exchanging efficiency needs to improve in hand with the increase in computing density and speed. But a shift from a scale-up (increasing total computing power through an increase in computing density) to a scale-out model (increase in total computing power by improving the performance of computer nodes at constant computing density) presents a challenge in implementing the fabric interconnecting a massive scale of computing elements at an acceptable level of power density and consumption [1].

The characteristics of HPC closely rely on internode communications, which means that network topology and routing algorithms greatly influence its performance. Typical interconnection topologies for HPC have been constantly proposed in the past few years, such as Fat-tree, Torus [2], and Dragonfly [3]. Fat-tree can perform well in massive data processing, but it is made of hierarchical standard switches that show high bandwidth redundancy and limited bandwidth scheduling flexibility. So, as the scale of a network increases, when trying to increase both the radix (port count) and the bandwidth per port of each switch, the resulting electronic core switch becomes too expensive to make, and its performance will be limited by scalability, reliability, and power consumption [4–6]. Compared with Fat-tree, the Torus network possesses some desirable features: lower cost for hardware, more achievable implementation, and lower latency [7,8]. In addition, routing algorithms [9,10], used for improving path diversity inside the interconnection network, are also more compatible with the Torus network. However, the critical drawback is the long routing path length of Torus architecture, which results in low routing efficiency and high latency. The Dragonfly is a two-level full-connecting network using high-radix switches that offers a low diameter and low network cost because it takes only three hops at most from one endpoint to another [3]. But doing so is at the expense of path diversity, which makes it vulnerable to certain adversarial traffic patterns [11].

Meanwhile, as the interconnection network is evolving towards larger scale and stronger performance, the current interconnection network based on electronic packet switching (EPS) has gradually exposed its limited flexibility and high end-to-end (ETE) delay, which limits the further expansion of the HPC network [12]. Therefore, in order to achieve lower

delay, the design of interconnection topologies with low diameter and low average shortest path length (both measured in hops) has become a primary requirement for a next-generation HPC system. The application of optical circuit switching (OCS) has made the low delay and high bandwidth topology a reality [13], and various interconnects [12,14,15] employing reconfigurable OCS in the core part of the fabric have been proposed. For example, IBM has demonstrated an OCS-based reconfigurable Dragonfly network for HPC systems composed of more than 13,000 nodes [16,17]. So the reconfigurable network architecture is an attractive solution that intelligently changes a network into a desirable topology.

In addition, HPC computing nodes regularly alternate computing/communication phases, resulting in bursty, uneven, and fluctuating traffic distribution. The traffic pattern of each HPC application contains valid information such as traffic distribution and an arrival model. Therefore, the communication intensity between different nodes in the form of traffic pattern is the most important information that we could use to efficiently allocate the bandwidth where it is needed. Current HPC systems carry a wide variety of services, and certain applications always show some specific regularity and perform one similar traffic pattern [12,13,18,19]. Furthermore, one HPC application that is run sometimes consumes entire system resources while calculations and data are exchanged [16]. Some previous studies show that a specific topology outperforms others under given traffic patterns due to some kind of "matching" between their particular computing node's communication characteristics and the topology structure [20–23]. Also, there are many specially designed HPC architectures for specific applications with certain communication patterns, e.g., "Anton," designed for biomolecular simulation [24]. Nevertheless, it is obvious that we could not build a general-purpose HPC architecture with fixed topology that suits all kinds of applications. Therefore, it is important to propose a reconfigurable interconnection architecture adaptable to various HPC applications with diverse traffic patterns.

In light of the above discussion, we design a hierarchical and reconfigurable optical/electrical interconnection network (HRIN) for HPC systems in order to accommodate various communication demands posed by different HPC applications. And we minimize the average path length (APL) of the unit communication intensity as the goal of topology optimization to reduce delays and improve throughput. This architecture implements a traffic pattern of an adaptive interconnection topology that uses compound graphs and a reconfigurable optical switch [25] to interconnect servers. Within each layer, the topology can be reconfigured according to its specific intralayer traffic patterns. For interlayer cross-connection, we use a conventional shuffle network topology to achieve the shortest path and multipath communication. In addition, we define the communication intensity matrix between every two nodes as a way to illustrate the traffic patterns. After identifying the traffic matrix, the target topology that needs to be reconfigured can be confirmed by the proposed reconfiguration algorithm.

The rest of this paper is organized as follows. Related work is discussed in Section 2. Section 3 presents the HRIN and the traffic pattern adaptive topology matching algorithm. In Section 4, we designed an interlayer cross-connection reconfiguration optimization algorithm and a routing algorithm. The simulation results are analyzed in Section 5. Finally, we summarize this work in Section 6.

2. RELATED WORK

In most traditional HPC systems, high-radix switches are used to interconnect lots of computing nodes (like Fat-tree) or each switch is only connected to adjacent switches (like Mesh), and they are smaller in diameter but require more switches. Popular regular topologies include k-ary n-cubes, whose degree is $2n$, such as Torus, Mesh, and Hypercube [26]. These topologies make a trade-off between degree and network diameter, and each switch connects to the same number of other switches or computing nodes. In fact, network topology has a great impact on the deployment of large HPC systems, especially in terms of cost and configuration.

The most important feature of optical switching is data rate transparency and ultrahigh bandwidth capacity, which will provide unprecedented data delivery capability for HPC. In [12,13], a two-network hybrid optical/electrical (O/E) interconnection network has been proposed: an OCS network handling long-lived elephant flow data delivery, while the EPS network carries aggregated mice flow data. The above work provides a more effective solution for HPC applications and illustrates the feasibility and potential performance advantages of the hybrid O/E interconnection network.

Meanwhile, according to different application types, the performance requirements of the interconnection topology will be different [27]. Therefore, the collection and tracking of data are critical in analyzing the different traffic patterns generated by different applications, and tools like Simgird [16] provide additional convenience along the way. IBM has developed a simulation tool, Venus [20], which provides pattern detection by collecting applications traces at the message passing interface (MPI) level. In [13,22,28–30], they have studied the architectural requirements, communication patterns, and traffic characteristics of various parallel scientific applications. They gathered performance data using a range of tools (subroutine profiling, MPI tracing, and accessing hardware counters). With this empirical data, the strongest conclusion of their study is that the applications they examined vary widely in their characteristics, and the performance of such applications is the final arbiter of an HPC system's utility. As a result, it appears that future HPC architectures will have to be extremely flexible and support multiple users and widely varying applications. We can conclude that the HPC traffic pattern has its own characteristics, and if we could detect the traffic pattern properly, we could reconfigure the interconnection topology into a desirable target topology with the purpose of improving network resources utilization.

Therefore, unlike the traditional topology or the interconnection design using high-radix routers/switches, our proposed architecture can implement multiple topology reconfiguration within the intralayer according to traffic requirements and realize the shortest path and multipath communication at the interlayer through a conventional single-stage shuffle network topology.

3. INTERCONNECT ARCHITECTURE

In the following sections, we will introduce the structure of the HRIN in detail.

Figure 1(a) shows an example of a HRIN comprising four intralayer reconfigurable architectures (which can be regarded as four clusters) and one interlayer cross-connection (bidirectional plane). From the perspective of simple physical wiring and local symmetry, we select the four nodes at the center of each layer as aggregation nodes (red nodes) and use high-speed channels (red solid lines) to achieve full connection. All aggregation nodes are connected to (gray solid lines and red solid arrow lines) the interlayer cross-connection, as shown in the circle on the left of Fig. 1(a). The remaining nodes (black nodes) are directly connected to EPS and OCS on each layer in Fig. 1(b). To make the figure clearer, we only show the complete link of one node within the intralayer in Fig. 1(b).

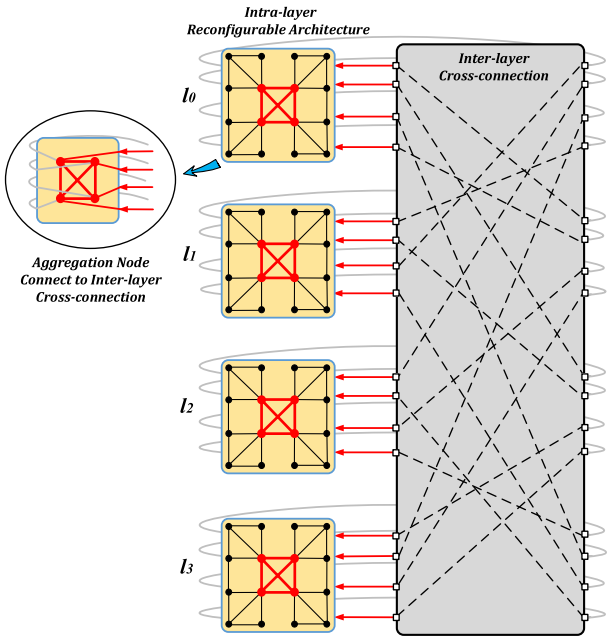

Fig. 1. HRIN. (a) Interlayer cross-connection, (b) intralayer reconfiguration architecture.

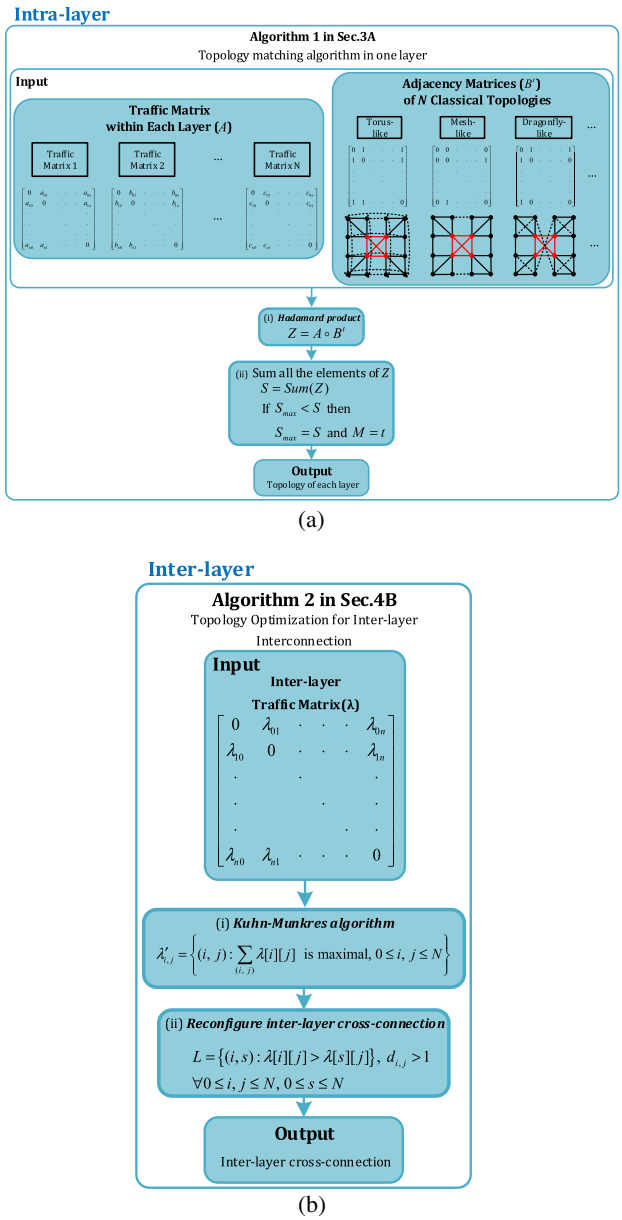

Fig. 2. Workflow of offline calculating topology. (a) Intralayer topology matching process, (b) interlayer cross-connection reconfiguration process.

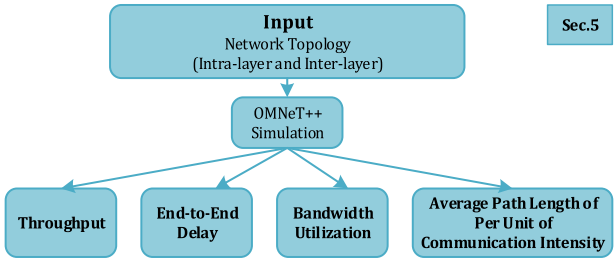

Fig. 3. Online simulation process.

Figures 2 and 3 are flow charts of network topology reconfiguration. The following sections will explain the entire process in detail. This section briefly introduces the two main steps:

Table 1. Parameters of Algorithm 1 and Algorithm 2

Parameters	Description
A	Traffic matrix within each layer
B	Adjacency matrices of N classical topologies
M and t	Number of the chosen topology
Z	Hadamard product of A and B^t
S	Sum of all the elements in Z
λ	Nonnegative traffic matrix
$\lambda'_{i,j}$	Set with the N most heavily communicating node groups
L	Set representing that source node groups need to be swapped by interlayer cross-connection
d	Hop counts between any of the source/destination node groups

Algorithm 1. Topology-Matching Algorithm in One Layer

Input:
A (traffic matrix of target pattern within each intralayer);
B^t (N classical topology adjacency matrices);
Output:
M (The number of the chosen topology)
Initialize $M = 0$
$S_{\max} = 0$ (the maximum of S)
for $t \in [1, N]$ **do**
$Z = A \circ B^t$ (*Hadamard product*)
$S = Sum(Z)$ (*Sum all the elements of* Z)
if $S_{\max} < S$ **then**
$S_{\max} = S$ and $M = t$
end if
end for
return M

Offline: Matching classical topology (similar to the typical topology, but with more or fewer links than a typical topology) for a target pattern within one layer and reconfiguring interlayer cross-connection. The intralayer topology matching process is discussed in Section 3.A [Fig. 2(a)]. The interlayer cross-connection reconfiguration process is discussed in Section 4.B [Fig. 2(b)].

Online: OMNeT++ simulation process in Section 5, compared to the performance of different topology types by different packet-generation rates (Fig. 3).

We have also added a table (Table 1) to illustrate the meaning of some letters (Algorithm 1 and Algorithm 2) in our paper.

A. Intralayer

In [16], we know that the performance of a specific topology is different under different traffic patterns. For a given traffic pattern of one layer, we can find the target topology matching the application through a topology matching algorithm (Algorithm 1).

A definition is introduced before Algorithm 1 is applied.

Definition: For two matrices A and B of the same dimension $m \times n$, the Hadamard product $A \circ B$ (or $A \odot B$) is a matrix of the same dimension as the operands, with elements given by

Algorithm 2. Interlayer Cross-Connection Optimization Algorithm

calculate $\lambda'_{i,j} = f(\lambda)$;
for $\forall\ (i, j) \in \lambda'_{i,j}$ **do**
if $d_{i,j} > 1$
for each $s : d_{s,j} = 1$ **do**
select $s : s \in L$, $\lambda[s][j] \leftarrow \lambda_{\min}[s][j]$;
if $\lambda[i][j] > T_L \cdot \lambda[s][j]$
$L \leftarrow (i, s)$;
end if
end for
end if
end for
return L

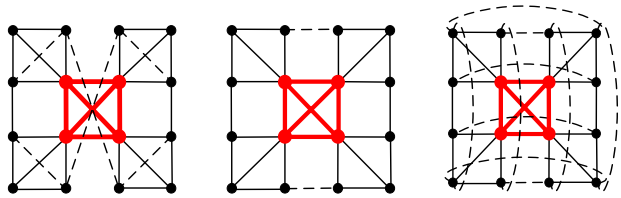

Fig. 4. Reconfigured classical topologies. (a) Dragonfly-like, (b) Mesh-like, (c) Torus-like.

$$(A \circ B)_{ij} = (A \odot B)_{ij} = (A)_{ij}(B)_{ij}.$$

For matrices of different dimensions ($m \times n$ and $p \times q$, where $m \neq p$ or $n \neq q$) the Hadamard product is undefined.

In Algorithm 1, we define A as a traffic matrix of the target pattern within each intralayer, (0, 1)-matrix B^t, ($t \in [1, N]$) as N classical topology adjacency matrices (in this paper, $N = 3$). We can calculate the matrix Z that is the Hadamard product of A and B^t and get the sum of all the elements in Z (S), of which the topology corresponding to the maximum value ($S_{\max}$) is the target topology.

Therefore, we simply interconnect the adjacent nodes in the intralayer to form a fixed circle (solid lines) as the basic public graph of Dragonfly-like, Mesh-like, and Torus-like nodes (see Fig. 4). These black nodes are then connected to the OCS switches and are reconfigured according to different traffic patterns. As shown in Fig. 1, at least one OCS switch is placed in each layer to realize topology reconstruction among 16 nodes. The four nodes located in the middle of the topology in each layer are selected as the aggregation node (red nodes) and use the high-speed channel (red solid line, 100G or beyond) for full connection to ensure that the cross-layer traffic can reach the destination node immediately and does not affect the layer's traffic. By connecting black nodes to the corresponding OCS switches and changing the switch cross-connection (black dotted line) according to different traffic patterns, we can realize the transformation of the network among the classical topologies.

B. Interlayer

Currently, the reconfiguration delay of commercial optical circuit switches is about milliseconds, which will have a great impact on their application performance during the logical topology transformation [1]. It is worth noting that, in order to reduce the bad impact of OCS reconfiguration, the interlayer cross-connection can reconfigure the topology for the interlayer traffic and realize multihop routing. Multihop routing could be employed to mitigate the overload problem when interlayer traffic demand goes beyond the capacity of a single interlayer link. In other words, aggregation nodes at each layer are programmed with the ability to switch packets from one layer directly back to optical switches in the optical domain. Thus, node groups with aggregation nodes as the core can switch packets received from the optical domain back to OCS switches again, and packets traverse multiple more times in the optical domain until they reach the destination. In this way, we can extend the excellent characteristics of the reconfigurable optical switching and effectively reduce network delay and increase network throughput.

In our topology, k is the radix of each aggregation node, n is the number of aggregation nodes, and o is the number of interlayer OCS switches. r represents the number of ports of each optical switch, as long as $n \cdot k \le o \cdot r$.

4. TOPOLOGY OPTIMIZATION FOR INTERLAYER INTERCONNECTION

In this section, we propose an interlayer cross-connection reconfiguration optimization algorithm and routing algorithm to better extend the excellent characteristics of reconfigurable optical switching.

A definition and a theorem are introduced before the Kuhn–Munkres algorithm and the interlayer cross-connection optimization algorithm are applied [31].

Definition: If the mapping $l: V(G) \to R$ satisfies that $\forall x \in X, y \in Y, l(x) + l(y) \ge \omega(x, y)$. The mapping l will be called the feasible labeling of the bipartite graph G:

$$E_l = \{xy | xy \in E(G), l(x) + l(y) = \omega(x, y)\}. \quad (1)$$

The spanning subgraph of the graph G with edge set of E_l is called an equal subgraph, denoted by G_l. The feasible labeling exists, for example,

$$\begin{cases} l(x) = \max\limits_{y \in Y} \omega(x, y), & x \in X \\ l(y) = 0, & y \in Y \end{cases}. \quad (2)$$

The feasible labeling is called the trivial labeling.

Theorem: The optimal distribution scheme of the equal subgraph G_l is also the optimal distribution scheme of the graph G.

A. Kuhn–Munkres Algorithm

Now, the Kuhn–Munkres algorithm [32–34] (the weighted bipartite graph maximum matching algorithm) will be introduced.

(i) The initial feasible labeling l is selected, and the equal subgraph G_l is determined. A distribution M is selected in the equal subgraph G_l.

(ii) The labeling in set X is totally allotted by the distribution M; then it stops. The distribution M is the optimal distribution scheme. Otherwise, the labeling u not allotted by the distribution M is selected in the equal subgraph G_l. Set $S = \{u\}$, $T = \emptyset$.

(iii) If $N_{G_l}(S) \supset T$, please go to (iv). If $N_{G_l}(S) = T$, take Eq. (3):

$$\alpha_l = \min_{x \in X, y \notin T} \{l(x) + l(y) - \omega(x, y)\}. \quad (3)$$

From Eq. (4), we can get $l = l'$, $G_l = G'_l$:

$$l'(v) = \begin{cases} l(v) - \alpha_l, & v \in S \\ l(v) + \alpha_l, & v \in T \\ l(v), & \text{others} \end{cases}. \quad (4)$$

(iv) A labeling y in the set $N_{G_l}(S) - T$ is selected. If the labeling y has been allotted by the distribution M, and $yz \in M$, then $S = S \cup \{z\}$, $T = T \cup \{y\}$. Please go to (iii). Otherwise, an augmentative path of the distribution M is selected from the equal subgraph G_l. Let it be $M = (M - E(P)) \cup (E(P) - M)$; please go to (ii).

B. Interlayer Cross-Connection Optimization Algorithm

Our interlayer cross-connection reconfiguration optimization algorithm (Algorithm 2) optimizes interlayer cross-connection by directly connecting high-communication intensity (heavily communicating) source/destination nodes groups (i.e., providing a one-hop connection), while the rest of the communication groups are connected with the multihop method. Figure 5 shows the Algorithm 2 process. Specifically, the nonnegative traffic matrix λ is given as the input, where the element represents the source/destination node pairs' communication intensity (as measured in the total number of transmitting packets), and the algorithm is mainly divided into two steps:

Step 1: This step obtains the set $\lambda'_{i,j}$ with the N most heavily communicating node groups:

$$\lambda'_{i,j} = \left\{ (i, j) : \sum_{(i,j)} \lambda[i][j] \text{ is maximal}, 0 \le i, j \le N \right\}. \quad (5)$$

We compute $\lambda'_{i,j}$ as defined in Eq. (5) by using an interlayer traffic matrix as input to solve the allocation problem on the source/destination node groups. This process is realized by the Kuhn–Munkres algorithm to obtain the maximum weight traffic-matching given the input.

Step 2: In the set $\lambda'_{i,j}$, if $\lambda_{i,j} > \lambda'_{i,j}$ and $d_{i,j} > 1$, where $d_{i,j} (0 \le i, j \le N)$ represents the hop counts between any of the source/destination node groups, then we can select a node group s, which is only a one-hop distance from the destination group j, from k alternative source node groups with the lowest communication intensity, as the new source to replace i.

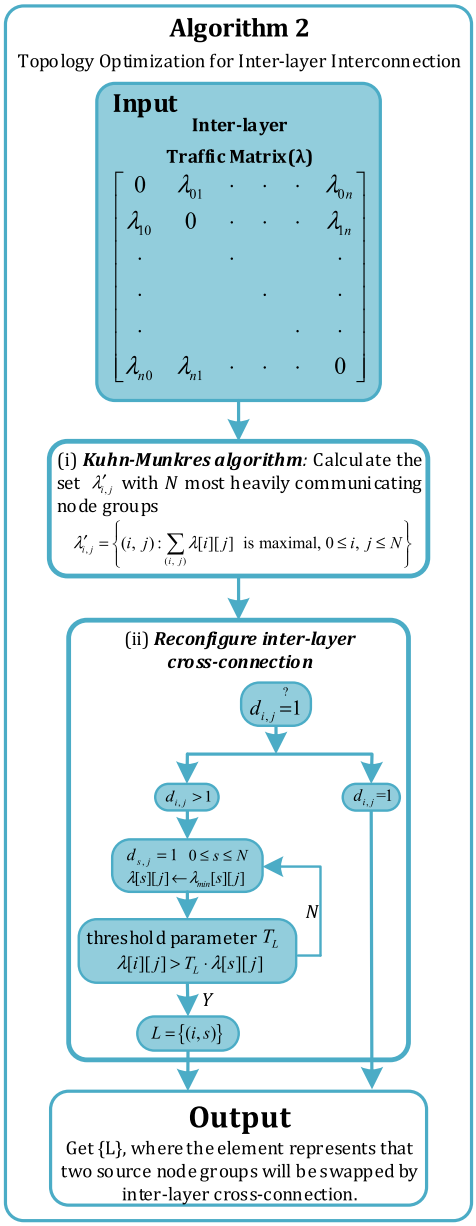

Fig. 5. Workflow of Algorithm 2.

After executing the two steps, we can obtain a set L, where the element (i, s) represents that two source node groups will be swapped by interlayer cross-connection:

$$L=\{(i,s):\lambda[i][j]>\lambda[s][j]\},\ d_{i,j}>1$$
$$\forall 0\le i,j\le N,\ 0\le s\le N. \tag{6}$$

Take the traffic matrix in Table 2 as an example. In Fig. 6(a), we depict the initial connection of the source/destination nodes and the flows $\lambda_{0,2}(i_0/i_2)$ and $\lambda_{3,1}(i_3/i_1)$. Note that both $\lambda_{0,2}$ and $\lambda_{3,1}$ require two hops to reach the destination [dashed line in Fig. 6(a)]. In this case, $\lambda'_{i,j}$ is calculated by Eq. (5):

$$\lambda'_{i,j}=\{(0,2),(3,1),(2,3),(1,0)\},$$

Table 2. Traffic Matrix (λ)

	Destination			
Source	**0**	**1**	**2**	**3**
0	0	536	968	181
1	107	0	356	115
2	281	325	0	510
3	89	865	56	0

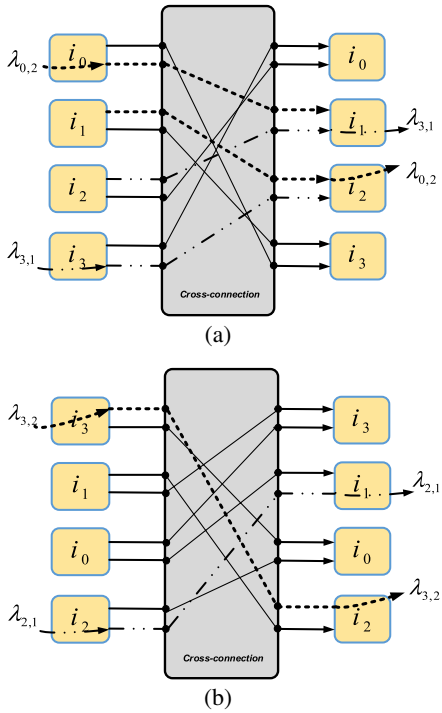

Fig. 6. (a) Cross-connection before optimization. (b) After reconfiguration, i_0, i_2, and i_3 need to be swapped with i_3, i_0, and i_2, respectively. In terms of routing, the traffic $\lambda_{0,2}$ and $\lambda_{3,1}$ in (a) are $\lambda_{3,2}$ and $\lambda_{2,1}$ in (b), both of which need only one hop to reach the destination.

with $\sum_{(i,j)}\lambda[i][j]=2450$. Then calculate the source nodes that need to be swapped to minimize the hop counts between the source/destination nodes by Step 2. The result is obtained by Eq. (6):

$$L=\{(0,3),(1,1),(2,0),(3,2)\}.$$

Elements in L represent that i_0, i_2, and i_3 need to be swapped with i_3, i_0, and i_2, respectively. The swap result is shown in Fig. 6(b). Note that nodes in the corresponding layer remain unchanged on the physical connection; it is only the routing tag generation entity that needs to be notified about node-swapping events after reconfiguring the interlayer cross-connection.

The interlayer cross-connection reconfiguration optimization algorithm is listed in Algorithm 2. The function $f(\lambda)$ on line number 1 represents an input interlayer traffic matrix Y, which returns an optimal allocation as the output of the Kuhn–Munkres algorithm (Step 1). Step 2 is executed on lines 2–11. In the process of algorithm execution, we set a

Algorithm 3. Floyd Algorithm

Input:
D (distance between source/destination node pairs);
$U_{B(u,v)}$ (used link bandwidth);
$R_{B(u,v)}$ (remaining link bandwidth);
Output:
P_D (the shortest path between (i, j));
1: **for** $0 \le i, j \le n, (u, v) \in (i, j)$ **do**
2: **calculate** $W(u, v) \leftarrow f_{(i,j)}(u, v)$
3: **if** $D_{(i,j)} > D_{(i,k)} + D_{(k,j)}$ **then**
4: $D_{(i,j)} = D_{(i,k)} + D_{(k,j)}$
5: $P_{D_{(i,j)}} = P_{(k,j)}$
6: **end if**
7: **end for**
8: **return** P_D

threshold parameter T_L as the constraint condition for selecting swapping node groups. Assuming $T_L = 1\%$, that is, if their communication intensity differs by more than 1%, the two node groups are swapped (line 6 in Algorithm 2). By setting constraints, unnecessary reconfiguration can be avoided when the node groups have the same degree of communication intensity.

C. Routing Algorithm

We optimized network routing using an improved Floyd algorithm (Algorithm 3). A definition and the Floyd algorithm are introduced before this algorithm is applied.

Definition: For a directed graph with n vertices, set an $n \times n$ matrix D^k with diagonal elements of 0, and other elements $D^k[i][j](i \neq j)$ represent the directed path length of (i, j), $k = 0, 1, 2, \ldots, n$:

(i) $D^0[i][j]$ represents the path length of the direct directed edges of (i, j), that is, the adjacency matrix Edge$[n][n]$.
(ii) $D^k[i][j]$ represents the shortest path length of (i, j) with the middle vertices' number not greater than k.
(iii) $D^n[i][j]$ represents the shortest path length of (i, j) finally obtained.

The goal of the Floyd algorithm is to find the shortest path between multiple sources in a given weighted graph. The algorithm assumes that $D[i][j]$ is the shortest path of (i, j). For each vertex's k, the algorithm checks whether $D[i][k] + D[k][j] < D[i][j]$ holds. If it holds, then set $D[i][j] = D[i][k] + D[k][j]$. In this way, when all vertices k are traversed, the shortest path length of (i, j) is recorded in $D[i][j]$.

Therefore, the recursive formula of the Floyd algorithm is

$$\begin{cases} D^0[i][j] = \text{Edge}[i][j] \\ D^k[i][j] = \min\{D^{k-1}[i][j],\ D^{k-1}[i][k] + D^{k-1}[k][j]\}, \\ \qquad\qquad k = 1, 2, \ldots, n. \end{cases}$$

We add link weight $W(u, v)$ in this algorithm, which is the ratio of bandwidth used and remaining on each link:

$$W(u, v) = \sum_{(i,j)\in V}^{K-1} \frac{U_{B(u,v)}}{R_{B(u,v)}}, \quad \forall (u, v) \in (i, j). \tag{7}$$

In Eq. (7), $W(u, v)$ represents the weight of link (u, v), $U_{B(u,v)}$ represents the bandwidth occupied on the link between (u, v), $R_{B(u,v)}$ is the remaining bandwidth of link (u, v), and V is the set of all source/destination node pairs. As $U_{B(u,v)}$ increases, $R_{B(u,v)}$ will decrease, and $W(u, v)$ will also increase. This link is the relative saturated link that needs to be avoided when selecting the shortest path.

5. SIMULATION AND ANALYSIS

We implemented our hierarchical and reconfigurable interconnection network on the OMNeT++ 5.4.1 simulator based on Ubuntu 18.04 and experimented with different topology types with different packet-generating rates in a static scenario.

OMNeT++ is an extensible, modular, component-based C++ simulation library and framework, primarily for building network simulators. OMNeT++ offers an Eclipse-based integrated development environment (IDE), a graphical runtime environment, and a host of other tools. OMNeT++ provides a component architecture for models. Components (modules) are programmed in C++, then assembled into larger components and models using a high-level language (NED) [35].

Before that, we declare the definition of throughput in our simulation. The throughput in the network or communication system refers to the average rate of successful data transmission over a communication channel (or a node) per unit time, usually in bits per second (bps). But in our simulation design, throughput cannot be measured accurately in bps, so we indirectly define the throughput as the total number of packets that the node successfully received. By selecting small-scale topologies (including 64 interconnected nodes) for simulations, we observed the throughput and average ETE delay of the network under different traffic patterns.

In simulations, the computing node number in each layer is set as 16, and the number of layers is four. Sixteen nodes are first connected to one $28 * 28$ intralayer EPS in each layer, of which four aggregation nodes are interconnected through high-speed channels, and the remaining 12 ordinary nodes are connected (not through high-speed channels) to adjacent aggregation nodes, respectively. Then 12 nodes are connected to one $20 * 20$ intralayer OCS in each layer because each node connects to no more than two nodes on intralayer reconfiguration architecture. The four aggregation nodes of each layer (a total of 16 aggregation nodes) are connected to one $16 * 16$ interlayer high-order OCS to achieve interlayer cross-connection. The OCS and EPS switch delays are considered as 20 and 400 ns, respectively, and the electronic packet channel bandwidth and optical transmission are considered as 10 Gbps in the intralayer, while the optical transmission rate is 100 Gbps in the interlayer. We set the packet size to a constant of 1024 bytes.

We compared the HRIN, the interconnection network based on EPS, and the reconfigurable O/E interconnection network of the same size. E stands for traditional electronic topology and O/E stands for optical/electrical interconnection topology in Figs. 7–9. As shown in Table 3, we selected

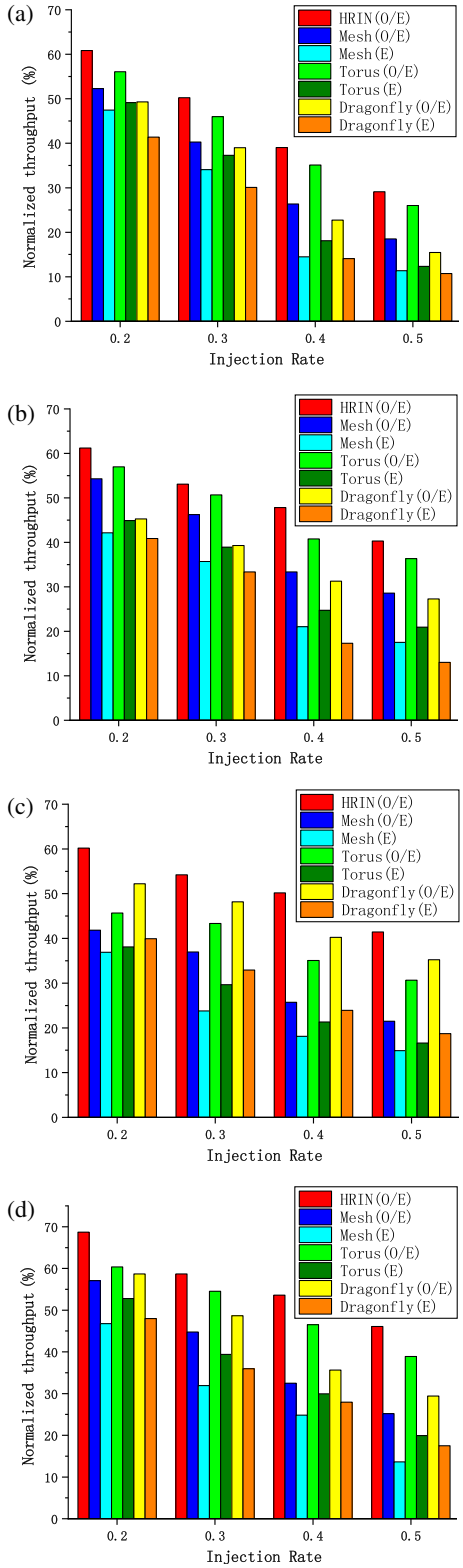

Fig. 7. Normalized throughput under different traffic patterns: (a) adversarial, (b) 2D neighbor exchange, (c) all-to-all (intralayer), (d) all-to-all (interlayer).

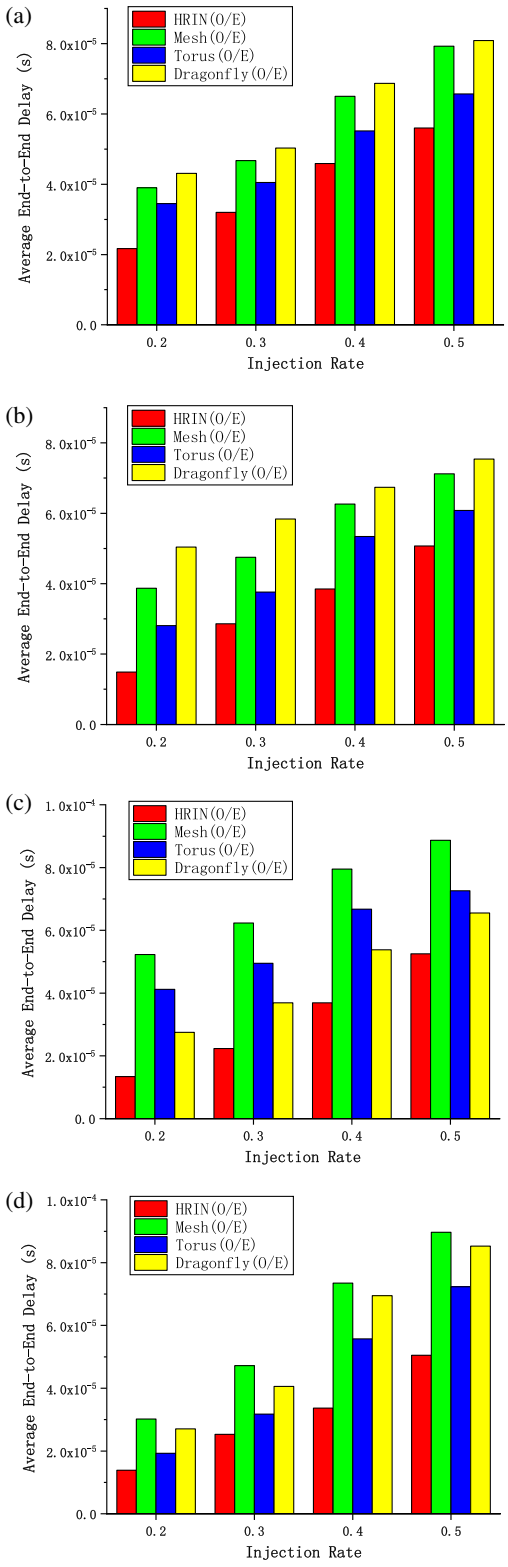

Fig. 8. Average ETE delay under different traffic patterns: (a) adversarial, (b) 2D neighbor exchange, (c) all-to-all (intralayer), (d) all-to-all (interlayer).

four different traffic patterns: adversarial (ADV) [20], 2D neighbor exchange, and all-to-all in interlayer and intralayer patterns [17].

The throughput is evaluated to measure the overall network capacity of the architecture. We can see that the throughput of each topology is different under different traffic patterns in

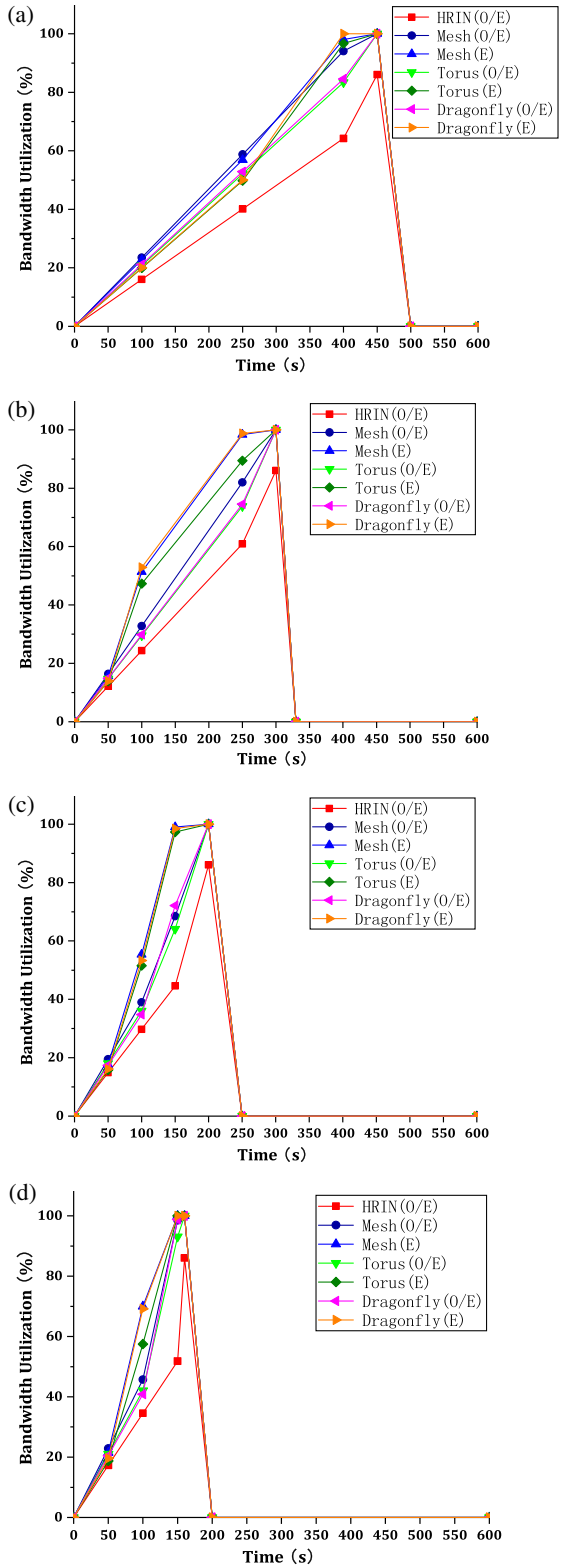

Fig. 9. Bandwidth utilization: (a) injection rates 0.2, (b) injection rates 0.3, (c) injection rates 0.4, (d) injection rates 0.5.

Fig. 7. In the traditional Mesh topology, the corresponding starting point and ending point in each dimension are connected to form the Torus topology, so the network diameter of Torus is relatively shorter and has a higher throughput than

Table 3. Mathematical Description of the Traffic Patterns

Traffic Pattern	Description
Adversarial	All the nodes in one layer send messages to nodes in the next layer.
2D neighbor exchange	50% of nodes in one layer send to neighbors within this layer; 50% to neighbors mapping in neighbor layers.
All-to-all in intralayer	Nodes communicate with others in the same layer over intralayer global channels.
All-to-all in interlayer	Nodes communicate with others in different layers over interlayer global channels.

Mesh in different patterns. Dragonfly is more suitable for all-to-all in an intralayer pattern in Fig. 7(c). We can also conclude that the reconfigurable O/E interconnection network improves the throughput by 10% at least by comparing the interconnection network based on EPS and the all-to-all in interlayer pattern; the throughput improvement for Torus(O/E) is higher in Fig. 7(d), about 19%. In Figs. 7(a) and 7(b), we note that as the injection rate increases, Torus(E) throughput is significantly lower than Dragonfly(O/E) (when it is more than 0.3) due to optical switching providing ultrahigh bandwidth capacity, which eases link congestion.

At the same time, the HRIN improves the throughput by 20% at least compared to an interconnection network based on EPS, and the throughput improvement for the HRIN is higher compared to Mesh(E) in the all-to-all in interlayer pattern in Fig. 7(d), about 33%. Compared with the reconfigurable O/E interconnection network, HRIN throughput improves by more than 10%. In Figs. 7(c) and 7(d), under all-to-all in intralayer and in interlayer patterns, the HRIN improves the throughput by more than 20% compared with Mesh(O/E), and more than 17% compared with Dragonfly(O/E) under ADV and 2D neighbor exchange patterns, in Figs. 7(a) and 7(b). Compared with Torus(O/E), the HRIN improves by 15% under the all-to-all in the intralayer pattern in Fig. 7(c).

Figure 8 shows the simulation results of ETE delay of the HRIN and the reconfigurable O/E interconnection network. Since the delay of the reconfigurable O/E interconnection network is significantly lower than the interconnection network based on EPS ($\geq 1.7 \times 10^{-4}$ s), we only compare four reconfigurable O/E interconnection networks. As shown in Figs. 8(a), 8(b), and 8(d), we observed that in three reconfigurable O/E interconnection networks, Torus(O/E) has lower latency than Mesh(O/E) and Dragonfly(O/E) in ADV, 2D neighbor exchange, and all-to-all in interlayer patterns. But we can see that Dragonfly(O/E) has the lowest latency compared with Mesh(O/E) and Torus(O/E) under the all-to-all in the intralayer pattern in Fig. 8(c). Although the ETE delay increases as the injection rate increases, we can see that the HRIN achieves the lowest latency, which is 28.8%, ~74.4% lower than Mesh(O/E); 14.8%, ~67.5% lower than Torus(O/E); and 19.9%, ~70.4% lower than Dragonfly(O/E).

In addition, the APL largely affects the network latency. A smaller network diameter with lower APL should be offered as a desired feature of the network design, enabling the HPC

Table 4. APL of Unit Communication Intensity of the HRIN and Traditional Classical Topologies

	HRIN (O/E)	Mesh (E)	Torus (E)	Dragonfly (E)
Adversarial	4.42	6.68	6.44	6.97
2D neighbor exchange	3.92	6.24	5.92	6.58
All-to-all in intralayer	3.87	8.51	7.38	8.02
All-to-all in interlayer	4.08	8.3	8.11	8.58

Table 5. APL of Unit Communication Intensity of the HRIN and Reconfigurable O/E Interconnecting Classical Topologies

	HRIN (O/E)	Mesh (O/E)	Torus (O/E)	Dragonfly (O/E)
Adversarial	4.42	5.28	5.15	5.54
2D neighbor exchange	3.92	4.7	4.54	4.83
All-to-all in intralayer	3.87	6.92	5.74	6.69
All-to-all in interlayer	4.08	6.58	5.81	6.98

to provide faster services. We define the APL of unit communication intensity, H_c [see Eq. (8)]. In Eq. (8), $c_{i,j}$ represents the communication intensity between source/destination nodes, and $h_{i,j}$ is the number of hop counts between source/destination nodes:

$$H_c = \frac{\sum_{0<i,j<n} c_{i,j} \cdot h_{i,j}}{\sum_{0<i,j<n} c_{i,j}}. \tag{8}$$

Tables 4 and 5 illustrate H_c by the HRIN and its competitors, Mesh, Torus, and Dragonfly, by applying different traffic patterns. It can be seen that the HRIN achieves the smallest APL of unit communication intensity, which is 16.3%, ∼54.5% smaller than Mesh; 13.7%, ∼49.7% smaller than Torus; and 18.8%, ∼52.4% smaller than Dragonfly. The smaller APL helps the HRIN achieve a lower network latency. Additionally, we can observe the APL of unit communication intensity of the reconfigurable O/E interconnection networks is significantly smaller than the APL of interconnection networks based on EPS because the optical switching technology increases the bandwidth of the link and further reduces the hop counts by the routing algorithm. At the same time, at the interlayer of the HRIN, we ensure that the hop counts of interlayers with high communication intensity are minimized by the Kuhn–Munkres algorithm, so that H_c of the HRIN is smaller than Mesh(O/E), Torus(O/E), and Dragonfly(O/E).

In addition, we also calculate the change of bandwidth utilization as the injection rate increases [see Eq. (9)]:

$$\eta = \frac{\sum_{i=0}^{63} S_i + \sum_{j=0}^{63} R_j}{2 \times \sum B} \times 100\%, \quad S = N_S \times n, \quad R = N_R \times n. \tag{9}$$

η indicates the bandwidth utilization. S indicates the total amount of data sent by the node. R indicates the total amount of data received by the node. B indicates the total of link bandwidth. N_S and N_R represent the number of packets sent and received by the node, respectively. n represents the size of a single packet. i and j represent source nodes and destination nodes, respectively.

Figure 9 shows bandwidth utilization at 0.2, 0.3, 0.4, and 0.5 (injection rates), respectively. The bandwidth utilization of the HRIN (O/E) is significantly lower than that of other topologies. Because the bandwidth interlayer is higher than the bandwidth intralayer, the HRIN (O/E) provides enough bandwidth for data transmission. As the injection rate increases, the peak of HRIN (O/E) bandwidth utilization is only 86.06%, not 100%. Conversely, as the injection rate increases, interconnected networks based on EPS cause more network congestion than reconfigurable O/E networks, resulting in bandwidth utilization reaching 100% earlier. In our simulation, we limited the communication strength of the node pairs, that is, the total amount of data sent. Therefore, the nodes are not in a continuous communication state in the network. With the total amount of data sent unchanged, as the injection rate increases, nodes will complete packets sent in advance, and the bandwidth utilization rate will be reduced to zero. When increasing the data traffic in the network, the size of the single packet, or injection rate, the bandwidth utilization will also increase to 100%; however, the bandwidth utilization reaches 100%, still later than several other topologies.

Therefore, we designed the HRIN, which has a great application prospect in the application of HPC systems. In the reconfigurable network with OCS switches, users can select the topology structure suitable for a specific traffic pattern in advance and conduct dynamic reconfiguration according to its characteristics so as to improve the flexibility of HPC systems.

6. SUMMARY

In this paper, we design a HRIN for an HPC system based on shuffle exchange topology. Furthermore, the reconfiguration optimization algorithm and routing algorithm help to enhance performance of the network. Simulation results prove the good performance of the HRIN with respect to the APL per unit of communication intensity, throughput, and network latency. Additionally, the results show that traffic patterns and topology reconfiguration studies are essential for optimizing reconfiguration strategies and improving network performance, which brings potential benefits to the HPC system. We will continue to improve our work by researching more flexible topologies, various traffic patterns, and routing algorithms.

Funding. National Key R&D Program of China (2018YFB1801702); National Natural Science Foundation of China (61771074); Outstanding Youth Scholars of China (61622102); Beijing University of Posts and Telecommunications; State Key Laboratory of Advanced Optical Communication Systems and Networks.

REFERENCES

1. D. Lugones, K. Katrinis, G. Theodoropoulos, and M. Collier, "A reconfigurable, regular-topology cluster/datacenter network using commodity optical switches," Future Gener. Comput. Syst. **30**, 78–89 (2014).
2. D. F. B. Garzón, C. G. Requena, M. E. Gómez, P. López, and J. Duato, "A family of fault-tolerant efficient indirect topologies," IEEE Trans. Parallel Distrib. Syst. **27**, 927–940 (2016).
3. L. Alawneh, A. Hamou-Lhadj, and J. Hassine, "Segmenting large traces of inter-process communication with a focus on high performance computing systems," J. Syst. Softw. **120**, 1–16 (2016).
4. B. Cheng, J. Fan, and X. Jia, "Parallel construction of independent spanning trees and an application in diagnosis on Möbius cubes," J. Supercomput. **65**, 1279–1301 (2013).
5. Z. Liu, J. Fan, and X. Jia, "Embedding complete binary trees into parity cubes," J. Supercomput. **71**, 1–27 (2015).
6. X. Wang, J. Fan, and C. Lin, "BCDC: a high-performance, server-centric data center network," J. Comput. Sci. Technol. **33**, 400–416 (2018).
7. M. Dhanak, P. D. Godbole, and R. A. Patil, "Torus network labeling in high performance computing," in *International Conference on Computing Communication Control and Automation* (2016).
8. T. Wang, Z. Su, and Y. Xia, "CLOT: a cost-effective low-latency overlaid torus-based network architecture for data centers," in *IEEE International Conference on Communications* (2015).
9. M. A. Wani and H. R. Arabnia, "Parallel edge-region-based segmentation algorithm targeted at reconfigurable multi-ring network," J. Supercomput. **25**, 43–63 (2003).
10. H. R. Arabnia, "Distributed stereocorrelation algorithm," Comput. Commun. **19**, 707–711 (1996).
11. B. Prisacari, G. Rodriguez, M. Garcia, E. Vallejo, R. Beivide, and C. Minkenberg, "Performance implications of remote-only load balancing under adversarial traffic in Dragonflies," in *Proceedings of the 8th International Workshop on Interconnection Network Architecture: On-Chip, Multi-Chip (INA-OCMC)* (ACM, 2014), article 5.
12. K. J. Barker, A. Benner, R. Hoare, A. Hoisie, A. K. Jones, D. K. Kerbyson, D. Li, R. Melhem, R. Rajamony, E. Schenfeld, S. Shao, C. Stunkel, and P. Walker, "On the feasibility of optical circuit switching for high performance computing systems," in *Proceedings of the ACM/IEEE Conference on Supercomputing (SC)* (2005), pp. 1–22.
13. K. Barker and D. Kerbyson, "Performance analysis of an optical circuit switched network for peta-scale systems," in *Proceedings of the Euro-Par Parallel Processing* (2007), pp. 858–867.
14. N. Farrington, G. Porter, S. Radhakrishnan, H. H. Bazzaz, V. Subramanya, Y. Fainman, G. Papen, and A. Vahdat, "Helios: a hybrid electrical/optical switch architecture for modular data centers," ACM SIGCOMM Comput. Commun. **41**, 339–350 (2010).
15. G. Wang, D. G. Andersen, M. Kaminsky, K. Papagiannaki, T. E. Ng, M. Kozuch, and M. Ryan, "c-Through: part-time optics in data centers," ACM SIGCOMM Comput. Commun. **40**, 327–338 (2010).
16. Y. Shang, B. Guo, W. Li, Y. Zhou, X. Li, Y. Zhang, and S. Huang, "Traffic pattern adaptive hybrid electrical and optical switching network for HPC system," IEEE Commun. Lett. **23**, 270–273 (2019).
17. C. Minkenberg, G. Rodriguez, B. Prisacari, L. Schares, P. Heidelberger, D. Chen, and C. Stunkel, "Performance benefits of optical circuit switches for large-scale dragonfly networks," in *Optical Fiber Communication Conference and Exhibition* (OSA, 2016), paper W3J.3.
18. S. Kamil, J. Shalf, L. Oliker, and D. Skinner, "Understanding ultra-scale application communication requirements," in *Proceedings of the IEEE International Workload Characterization Symposium* (2005), pp. 178–187.
19. Y. Hu and M. Koibuchi, "Optimizing slot utilization and network topology for communication pattern on circuit-switched parallel computing systems," IEICE Trans. Inf. Syst. **E102-D**, 247–260 (2019).
20. Y. Shang, B. Guo, W. Li, Y. Zhou, X. Li, Y. Zhang, and S. Huang, "Optical circuit switching enabled reconfigurable HPC network for traffic pattern," in *Optical Fiber Communication Conference* (2018), paper W4I.5.
21. M. Koibuchi, I. Fujiwara, K. Ishii, S. Namiki, F. Chaix, H. Matsutani, H. Amano, and T. Kudoh, "Optical network technologies for HPC: computer-architects point of view," IEICE Electron. Express **13**, 20152007 (2016).
22. R. Cypher, A. Ho, S. Konstantinidou, and P. Messina, "Architectural requirements of parallel scientific applications with explicit communication," in *Proceedings of the 20th Annual International Symposium on Computer Architecture* (1993), pp. 2–13.
23. L. Oliker, A. Canning, J. Carter, C. Iancu, M. Lijewski, S. Kamil, J. Shalf, H. Shan, E. Strohmaier, S. Ethier, and T. Goodale, "Scientific application performance on candidate petascale platforms," in *Proceedings of the International Parallel & Distributed Processing Symposium (IPDPS)* (2007), pp. 1–12.
24. D. E. Shaw, M. M. Deneroff, R. O. Dror, J. S. Kuskin, R. H. Larson, J. K. Salmon, C. Young, B. Batson, K. J. Bowers, and J. C. Chao, "Anton, a special-purpose machine for molecular dynamics simulation," ACM SIGARCH Comput. Archit. News **35**(2), 1–12 (2007).
25. D. P. Agrawal, C. Chen, and J. R. Burke, "Hybrid graph-based networks for multiprocessing," Telecommun. Syst. **10**, 107–134 (1998).
26. M. Koibuchi, H. Matsutani, H. Amano, D. F. Hsu, and H. Casanova, "A case for random shortcut topologies for HPC interconnects," in *Annual International Symposium on Computer Architecture (ISCA)* (IEEE, 2012), pp. 177–188.
27. F. Chaix, I. Fujiwara, and M. Koibuchi, "Suitability of the random topology for HPC applications," in *Proceedings of the Euromicro International Conference on Parallel, Distributed, and Network-Based Processing (PDP)* (2016), pp. 301–304.
28. J. Vetter and A. Yoo, "An empirical performance evaluation of scalable scientific applications," in *Proceedings of the ACM/IEEE Conference on Supercomputing* (2002).
29. K. Wen, P. Samadi, S. Rumley, C. P. Chen, Y. Shen, M. Bahadori, K. Bergman, and J. Wilke, "Flexfly: enabling a reconfigurable dragonfly through silicon photonics," in *SC16: International Conference for High Performance Computing, Networking, Storage and Analysis* (2016), pp. 166–167.
30. C. Minkenberg and G. R. Herrea, "Trace-driven co-simulation of high-performance computing systems using OMNeT++," in *Proceedings of the International Workshop on Omnet++* (2009).
31. J. H. Yin and K. Y. Wu, *The Graph Theory and Its Algorithm* (China Science and Technology, 2005).
32. J. Munkres, "Algorithms for the assignment and transportation problems," J. SIAM Control **5**, 32–38 (1957).
33. H. W. Kuhn, "Variants of the Hungarian method for assignment problems," Nav. Res. Logist. Q. **3**, 253–258 (1956).
34. H. W. Kuhn, "The Hungarian method for the assignment problem," Nav. Res. Logist. Q. **2**, 83–97 (1955).
35. A. Varga, "The OMNeT++ discrete event simulation system," in *Proceedings of the European Simulation Multiconference (ESM)*, Prague, Czech Republic (2001).

Zuoqing Zhao is currently pursuing the Ph.D. degree at the School of Optoelectronic Information, Institute of Information Photonics and Optical Communications, Beijing University of Posts and Telecommunications, Beijing, China. His research interest is optical interconnection networks in high-performance computing.

Bingli Guo (M'11) received the B.S. degree in telecommunication engineering from Jilin University, Jilin, China, in 2005, and the Ph.D. degree in electromagnetic field and microwave technology from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2011. From 2011 to October 2013, he was a researcher with Peking University, Beijing. From November 2013 to September 2015, he was a research associate with the High Performance Network Group, University of Bristol, Bristol, U.K. He is currently an Associate Professor with BUPT. His research interests include software-defined networking-enabled data center networks and OTDM switching.

Yu Shang received the B.S. degree in communication engineering from Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2015. She is currently working toward the Ph.D. degree at the State Key Laboratory of Information Photonics and Optical Communications, BUPT, and she is a visiting student with the University of California, Davis, Davis,

CA. Her current research interests include optical interconnection networks in data centers and high-performance computing.

Shanguo Huang (M'09) received the Ph.D. degree from Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2006. He is currently a Professor with the State Key Laboratory of Information Photonics and Optical Communications, and the Deputy Dean with the School of Science, BUPT, Beijing. He has been actively undertaking several national projects, has published three books and more than 150 journal articles and refereed conferences, and authorized 18 patents. His research interests include network designing, planning, traffic control, and resource allocations, especially network routing algorithms and performance analysis. He was awarded the Beijing Higher Education Young Elite Teacher, the Beijing Nova Program, the Program for New Century Excellent Talents in University from the Ministry of Education in 2011–2013, and the National Science Fund for Excellent Young Scholars in 2016.