Designing High-Performance Interconnection Networks with Host-Switch Graphs

Ryota Yasudo, Michihiro Koibuchi, *Member, IEEE,* Koji Nakano, *Member, IEEE Computer Society,* Hiroki Matsutani, *Member, IEEE,* and Hideharu Amano, *Member, IEEE*

Abstract—This paper aims at establishing a method for designing high-performance network topologies to bridge a gap between theoretical and practical studies. To this end, we present a novel graph called a host-switch graph, which consists of host vertices and switch vertices with maximum degree 1 and r, respectively. This graph represents a network topology of a practical parallel/distributed computer system with host computers connected by r-port switches. We discuss important metrics for designing high-performance interconnection networks: the host-to-host average shortest path length (h-ASPL) and the bisection width (BiW). In particular, we explore a method for constructing host-switch graphs with low h-ASPL and high BiW that connect the fixed number of hosts via any number of r-port switches. We demonstrate that the number of switches that provides the minimum h-ASPL can mathematically be approximated, and the minimum number of switches that provides a certain BiW can experimentally be approximated. On the basis of the approximations, we propose a randomized algorithm for searching host-switch graphs. We then apply the graphs to interconnection networks and compare them with typical network topologies. As compared with the torus, the dragonfly, and the fat-tree, our networks attain higher performance and smaller power and costs.

Index Terms—Network topology, interconnection network, average shortest path length, bisection width, optimization.

1 Introduction

1.1 Theoretical Studies of Interconnection Networks

THEORETICALLY, a topology of a computer network is represented as an undirected graph, in which vertices and edges correspond to computers and communication links, respectively. The performance potentiality of the network can be measured by analysing topological properties of the graph. In the design of networks for computer systems such as multiprocessors and supercomputers, there are certain requirements and limitations. In particular, requirements include the number of nodes, and limitations include the degree and the diameter. Hence the three parameters above have been studied in graph theory. The degree/diameter problem (DDP) is a classical problem for such studies. The DDP is the problem of finding the largest number of vertices in a graph of given maximum degree Δ and diameter D. The known upper bound—called the Moore bound [1]—on the number of vertices of an undirected graph is $1 + \Delta \sum_{i=0}^{D-1} (\Delta - 1)^i$. Near-optimal/optimal solutions of the DDP are considered for topologies of interconnection networks [2]–[4].

However, the DDP solutions may not be usable for building network topologies in practical interconnection networks. This is because the DDP requires the specific number of vertices, and hence we cannot meet technical requirements such as the number of nodes. To cover this shortcoming, we should fix the number of vertices (*order*) of a graph. We can consider the order/degree problem (ODP), the problem of finding the smallest diameter in a graph of given order and degree. Although less attention is given to the ODP as compared with the DDP, the ODP is recently studied by designers of interconnection networks [5].

In the field of network science, researchers find that complex networks such as social networks provide low diameter and ASPL. Thus some models are proposed, e.g, a cycle plus a random matching [6], the Erdős-Rényi model (random graph) [7], and the Watts-Strogatz model (small-world networks) [8]. Some solutions for the ODP are such complex graphs and applied to computer systems, including high-performance computing systems [9], data centers [10], and on-chip networks [11]. To apply such complex topologies to practical networks, physical layouts [12] and routing algorithms [13] are also studied.

Even if we tackle the ODP, however, a shortcoming remains; in conventional graph theory, one kind of vertex is considered on a graph, though two types of nodes—hosts and switches—exist in typical interconnection networks. Hence, the mapping between vertices and physical devices is not obvious. If we regard vertices as switches, we have no information for hosts. This is a serious issue because the mapping strongly affects the network performance (we show this in Section 4). Therefore, we should radically change both a model of interconnection networks and a graph problem.

1.2 Practical Studies of Interconnection Networks

Practical studies on topologies of interconnection networks for parallel/distributed computer systems have a long

- *R. Yasudo, H. Matsutani, and H. Amano are with Keio University, 3–14–1, Hiyoshi, Kohoku-ku, Yokohama, JAPAN 223–8522.*
  E-mail: yasudo@am.ics.keio.ac.jp, matutani@arc.ics.keio.ac.jp, hunga@am.ics.keio.ac.jp.
- *M. Koibuchi is with National Institute of Informatics, 2–1–2, Hitotsubashi, Chiyoda-ku, Tokyo, JAPAN 101–8430.*
  E-mail: koibuchi@nii.ac.jp.
- *K. Nakano is with Hiroshima University, 1–4–1, Kagamiyama, Higashihiroshima-shi, Hiroshima, JAPAN 739–8527.*
  E-mail: nakano@cs.hiroshima-u.ac.jp.

history. In the 1970s, hypercubes were used in many systems such as Cosmic Cube [14]; in the 1980s, 2-D/3-D tori and meshes became the mainstream due to their short cables that provide high bandwidth and cost-efficiency; from the 1990s to 2000s, as the number of nodes becomes over 10 thousand, high-radix networks such as the dragonfly [15] are researched for reducing communication overhead; and now, in the 2010s, the high-radix networks are used in commercial high-performance computers [16], [17].

All the networks above are *direct networks*, which denote the networks such that a certain number of hosts are connected to each switch. In addition to direct networks, *indirect networks* are also used, which denote the networks such that some switches are connected with a certain number of hosts while the other switches are connected with no hosts. Above all, the fat-tree [18] is widely used in parallel/distributed computer systems from generation to generation, though technology for each generation is different (e.g., both CM-5 [19] in the 1980s and Tianhe-2 [20] in the 2010s use the fat-tree). In this respect, indirect networks contrast with direct networks. For this reason, the question of our interest is how we should uniformly discuss direct and indirect networks (note that prior theoretical study based on the DDP and the ODP deals with only direct networks). This should be studied systematically, but there has been no prior research to answer this question yet. Also, the rationality of existing topologies should be backed by graph theory.

1.3 Our Concept and Contribution

The study set forth in this paper aims at establishing a novel method for designing high-performance network topologies to bridge a gap between theoretical studies based on graph theory and practical studies based on computer engineering. To this end, we present a novel graph called a *host-switch graph*, which consists of *host* vertices and *switch* vertices (Fig. 1). A host can be connected to exactly one switch using an edge. A switch can be connected to at most r vertices, each of which is a host or a switch. Clearly, a host-switch graph represents a topology of a computer network with 1-port host computers and r-port network switches. Thus, studying the topological characteristics of host-switch graphs leads to find good topologies for practical computer systems.

In this paper, we deal with two topological properties that are important for designing interconnection networks, the host-to-host average shortest path length (h-ASPL) and the bisection width (BiW). We propose a method for designing a topology with low h-ASPL and high BiW. By analysing host-switch graphs, we provide answers to the following questions: (1) *given the number of hosts and the number of ports per switch, how many switches should be used?*; and (2) *which is better, direct or indirect networks, in terms of the h-ASPL and the BiW?*

1.4 Structure of the Paper

First, Section 2 provides theoretical foundation of host-switch graphs; we formally define a host-switch graph and provide upper and lower bounds on the maximum number of hosts, the diameter, and the h-ASPL. Second, Sections 3-4 present host-switch graphs with low h-ASPL; we take deterministic and heuristic approaches in each section.

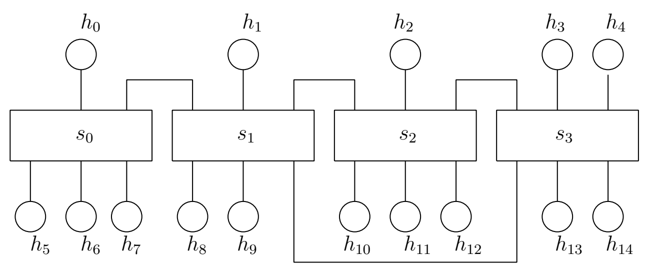


Fig. 1. An example of a host-switch graph ($n = 15, m = 4, r = 6$).

Here we demonstrate that the heuristic approach is more practical than the deterministic one for certain reasons. We empirically show that the optimal number of switches is a key parameter for host-switch graphs in terms of the h-ASPL and also the BiW. Third, in Section 5, we practically compare proposed network topologies with existing ones in terms of performance, topological properties, power consumption, and cost breakdowns. Finally, we conclude the paper in Section 6.

2 A Host-Switch Graph

2.1 Definition and Notation

A *host-switch graph* is a 3-tuple $G = (H, S, E)$ with integer parameters $n \geqslant 3$, $m \geqslant 1$, and $r \geqslant 3$ where

- $H = \{h_0, h_1, \ldots, h_{n-1}\}$ is a set of n elements called *host vertices* (or simply *hosts*),
- $S = \{s_0, s_1, \ldots, s_{m-1}\}$ is a set of m elements called *switch vertices* (or simply *switches*), and
- $E = \{\{s_i, s_j\} \mid s_i, s_j \in S\} \cup \{\{h_i, s_j\} \mid (h_i \in H) \wedge (s_j \in S)\}$ is a set of elements called *edges*.

The number n of hosts is called the *order* of G. Each host must be connected with exactly one edge while each switch is connected with at most r edges. Thus each switch must have at least r ports. The number r of required ports per switch is called the *radix* of G. In Fig. 1 we illustrate an example of a host-switch graph with 15 hosts and 4 switches with radix $r = 6$. Throughout this paper, a circle and a rectangle represent a host and a switch, respectively.

Clearly, at least $m - 1$ edges are necessary to connect m switches such that they are reachable each other. Since m switches can connect at most mr edges, we can state:

Lemma 1 (Trivial upper bound on the order) *For any connected host-switch graph with m r-port switches, the order n is not greater than $mr - 2(m - 1)$.*

For any two hosts h_i and h_j, let $\ell(h_i, h_j)$ denote the number of edges along the shortest path between h_i and h_j. For example, $\ell(h_0, h_{14})$ of a host-switch graph shown in Fig. 1 is 4, because the shortest path between them is $(h_0, s_0, s_1, s_3, h_{14})$. Using $\ell(h_i, h_j)$, we can define two topological properties. The *diameter* $D(G)$ of a host-switch graph is defined as $\max\{\ell(h_i, h_j) \mid 0 \leqslant i < j < n\}$. The *host-to-host average shortest path length (h-ASPL) $A(G)$* is defined as $\sum_{0 \leqslant i < j < n} \ell(h_i, h_j) / \binom{n}{2}$. These metrics are essentially different from the diameter and the average shortest path length (ASPL) of an ordinary undirected graph in that the considered path is between hosts rather than switches. In this paper we mainly discuss the h-ASPL, because it measures the ideal all-to-all communication latency of interconnection networks.

2.2 Upper and Lower Bounds

Let us consider tight upper bound on the order of a host-switch graph with r and $D(G)$. For any source host $h_s \in H$, we can partition all the hosts in H into subsets $H_0, H_1, \ldots$ such that $H_i = \{h_d \in H \mid \ell(h_s, h_d) = i\}$. Similarly, we can partition all the switches in S into subsets $S_1, S_2, \ldots$ such that $S_i = \{s_d \in S \mid \ell(h_s, s_d) = i\}$. Let $A_{h_s}(G)$ and $D_{h_s}(G)$ denote a single-source h-ASPL from h_s and a single-source diameter from h_s, respectively. Using these notations, we obtain the upper bound on the order, as follows:

Theorem 1 (Upper bound on the order) *For any host-switch graph with radix r and diameter $D(G)$, the order n of a host-switch graph is not greater than $(r-1)^{D(G)-1}+1$.*

Proof: For any fixed host h_s of a host-switch graph, let N_i be the upper bound on $|H_i| + |S_i|$. Clearly, N_i is equal to

$$\begin{cases} |H_0| = 1, & \text{if } i = 0 \\ |S_1| = 1, & \text{if } i = 1 \\ |S_{i-1}|\,(r-1)\,. & \text{if } i > 1 \end{cases} \tag{1}$$

Hence, to maximize the order, we must satisfy $N_i = |S_i|$ for $1 \leqslant i < D(G)$ and $N_i = |H_i|$ for $i = D(G)$. In this situation, the order is

$$\sum_{i=0}^{D(G)} |H_i| = (r-1)^{D(G)-1} + 1.$$

□

The lower bound on the diameter follows from Theorem 1:

Corollary 1 (Lower bound on the diameter) *For any host-switch graph with order n and radix r, the diameter is not less than $\lceil \log_{r-1}(n-1) \rceil + 1$.*

Let us call a host-switch graph with a root host h_s and $(r-1)^{D_{h_s}(G)-1}$ leaf hosts a *full host-switch tree*. Clearly, the lower bound on $A_{h_s}(G)$ is the lower bound on $A(G)$.

We define a *complete host-switch tree* as follows:

1) A full host-switch tree is a complete host-switch tree.
2) A host-switch tree obtained by performing the following operations to any complete host-switch tree T is also a complete host-switch tree: (A) if T has a switch connected to less than r vertices, then connect a new host to it; and (B) if T has no such switch, then we pick one of the hosts closest to h_s and replace it by a new switch with two hosts.

Both operations (A) and (B) increase the order n by one, and hence a complete graph host-switch tree is a full host-switch tree if and only if n is equal to $(r-1)^{d-1}+1$ for some d.

In a complete host-switch tree T with a root host h_s, $H_{D_{h_s}(T)} \cup H_{D_{h_s}(T)-1}$ includes all the leaf hosts. Clearly, a complete host-switch tree has at least one host in $H_{D_{h_s}(T)-1}$ if it is not a full host-switch tree. Also, at most one switch in $S_{D_{h_s}(T)-1}$ in a complete host-switch tree can take degree less than r.

Now we can state the following theorem:

Theorem 2 (Lower bound on the h-ASPL) *A single-source h-ASPL from the root of a complete host-switch tree provides the lower bound on the h-ASPL.*

Proof: Consider any host-switch tree T with a root host h_s. If there exists a host $h_a \in H_i$ $(1 \leqslant i \leqslant D_{h_s}(T) - 2)$, then we can decrease $A_{h_s}(T)$ by performing the following operations: (1) replace h_a with a new switch s_m connected with a host; (2A) if $r > 3$, then reconnect more than two hosts in $H_{D_{h_s}(T)}$ to s_m; and (2B) if $r = 3$, then reconnect two hosts in $H_{D_{h_s}(T)}$ and replace a switch in $S_{D_{h_s}(T)-1}$ with another host in $H_{D_{h_s}(T)}$. Hence, a host-switch tree can provide the lower bound on $A_{h_s}(T)$ only if H_i is empty for all i $(1 \leqslant i \leqslant D_{h_s}(T) - 2)$; in this case, $A_{h_s}(T)$ takes the minimum value if and only if a host-switch tree is a complete host-switch tree. Therefore a complete host-switch tree provides the lower bound on the single-source h-ASPL, which is also the lower bound on the h-ASPL. □

By Theorem 2, we can compute the lower bound on the h-ASPL as follows:

$$\begin{cases} D^-, & \text{if } n = (r-1)^{D^- - 1} + 1 \\ D^- - \alpha/(n-1), & \text{otherwise} \end{cases}$$

where $D^- = \lceil \log_{r-1}(n-1) \rceil + 1$ is the lower bound on the diameter of G, and α denotes the number of hosts in $H_{D_{h_s}(T)-1}$ in a complete host-switch graph T. In $H_{D_{h_s}(T)-1}$, at most $(r-1)^{D^- -2}$ hosts can be connected. However, it is less than n by $(n - 1 - (r-1)^{D^- -2})$ if T is not a full host-switch tree, and hence we must run operations (B) and then (A). After we run operation (B), we can increase the number of hosts by $r-2$ (remove one host from $H_{D_{h_s}(T)-1}$ and add $r-1$ hosts to $H_{D_{h_s}(T)}$). Thus, we have

$$\alpha = (r-1)^{D^- -2} - \left\lceil \frac{(n - 1 - (r-1)^{D^- -2})}{(r-2)} \right\rceil .$$

3 Deterministic Construction of Host-Switch Graphs

In this section we discuss host-switch graphs that can deterministically be constructed. Here, the relationship between the diameter and the order is easy to analyse. We thus discuss the *radix/diameter problem (RDP)*, which is similar to a classical problem called the degree/diameter problem [1], [21]. The RDP is defined as follows:

Problem 1 (Radix/Diameter Problem) *Given natural numbers r and D, find a host-switch graph with radix r and diameter D that can connect the largest possible number of hosts.*

The upper bound for this problem is given by Theorem 1.

3.1 Host-Switch Graphs of Diameter 2

Since two hosts are connected via at least one switch, the shortest path lengths between any pair of hosts are at least 2. We thus begin with host-switch graphs of diameter 2. In this case, all the hosts must be connected with a single switch, and thus we can state:

Theorem 3 *A host-switch graphs of diameter 2 can connect at most r hosts.*

3.2 Host-Switch Graphs of Diameter 3

In a host-switch graph of diameter 3, every path must be either host-switch-host or host-switch-switch-host. Thus the m switches in the host-switch graph must constitute a clique (complete graph). Let us call such a graph a *clique host-switch graph* (or specifically an *m-switch clique host-switch graph*). Each switch of a clique host-switch graph must be connected with $m-1$ switches, and hence the host-switch graph can be connected with at most $r-m+1$ hosts. Thus, we can state:

Lemma 2 *A host-switch graph can take diameter 3 only if $m \leqslant r+1$ and $n \leqslant m(r-m+1)$.*

Since $\partial m(r-m+1)/\partial m = -2m+r$, the order n is maximized when $m=(r+1)/2$ if r is odd and $m=r/2$ or $m=r/2+1$ if r is even. Thus, we can state:

Theorem 4 *A host-switch graph of diameter 3 can connect at most $(r+1)^2/4$ hosts if r is odd and $r(r+2)/4$ hosts if r is even.*

3.3 Host-Switch Graphs of Diameter 4

3.3.1 Biclique Host-Switch Graph

A typical host-switch graph of diameter 4 is a host-switch graph with the switches that constitute a complete bipartite graph (a.k.a. biclique). Let us call such a graph a *biclique host-switch graph*. Let K_{m_1,m_2} denote a biclique host-switch graph such that the switches constitute a biclique $G=(V_1,V_2,E)$ with $|V_1|=m_1$ and $|V_2|=m_2$; then m is equal to m_1+m_2. In Fig. 2a we illustrate an example of a biclique host-switch graph. Since any switch must be connected with all the switches in the other subset, we can state:

Lemma 3 *Any biclique host-switch graph satisfies $m_1 \leqslant r$, $m_2 \leqslant r$, and $m < 2r$.*

The maximum number $n_{\max}$ of hosts connected with K_{m_1,m_2} is

$$m_1\left(r-m_2\right)+m_2\left(r-m_1\right)=r\left(m_1+m_2\right)-2m_1m_2. \quad (2)$$

Thus, the increment of m_1 by 1 induces the increment of $n_{\max}$ by $r-2m_2$. Let $\Delta_{n_{\max}}$ be $r-2m_2$. Let us discuss the value of $n_{\max}$ in the following cases.

Case 1: $m_2 = r/2$**.**
In this case $\Delta_{n_{\max}}$ is equal to 0, and thus $n_{\max}$ is constantly $r^2/2$ regardless of the value of m_1.

Case 2: $m_2 < r/2$**.**
In this case $\Delta_{n_{\max}}$ is greater than 0. Thus $n_{\max}$ is maximized when $m_1=r$, and then $n_{\max}$ becomes r^2-m_2r. This value is maximized when $m_2=1$. Consequently, $n_{\max}$ is at most $r(r-1)$ in this case.

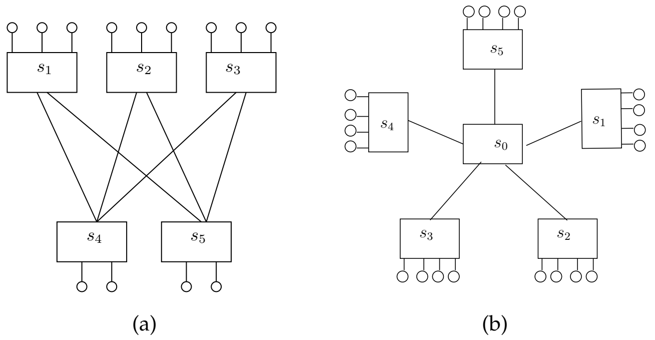


Fig. 2. Examples of a biclique host-switch graph with $r=5$; (a) $\{m_1,m_2\}=\{3,2\}$, $n=13$ and (b) star host-switch graph with $n=20$.

Case 3: $m_2 > r/2$**.**
In this case $\Delta_{n_{\max}}$ is less than 0. Thus $n_{\max}$ is maximized when $m_1=1$, and then $n_{\max}$ becomes $r+m_2(r-2)$. Since r is more than 2 from the definition of a host-switch graph, $n_{\max}$ is maximized when $m_2=r$. Consequently, $n_{\max}$ is at most $r(r-1)$ in this case.

Since $r^2/2$ is less than $r(r-1)$, we can state:

Theorem 5 *A biclique host-switch graph can connect at most $r(r-1)$ hosts if $\{m_1,m_2\}=\{r,1\}$.*

We name a biclique host-switch graph with $\{m_1,m_2\}=\{r,1\}$ a *star host-switch graph* after a star network [22]. In Fig. 2b we illustrate an example of a star host-switch graph.

3.3.2 XY-Clique Host-Switch Graph

Suppose that m switches are arranged in a $\sqrt{m}\times\sqrt{m}$ 2-dimensional grid. Every switch is connected to other switches in the same column and in the same row. We call the host-switch graph above an *XY-clique host-switch graph*. In Fig. 3 we show an example of an XY-clique host-switch graph. Each switch is connected with $2(\sqrt{m}-1)$ switches, and consequently it can be connected with at most $r-2\sqrt{m}+2$ hosts. Thus, we can state:

Lemma 4 *An XY-clique host-switch graph can be constructed if and only if $m<(r+2)^2/4$ and $n \leqslant m(r-2\sqrt{m}+2)$.*

Since $\partial(m(r-2\sqrt{m}+2))/\partial m = -3\sqrt{m}+r+2$, the value of n is maximized when $m=(r+2)^2/9$. Thus, assuming $m\in\mathbb{Q}$, we can state that an XY-clique host-switch graph can connect at most $(r+3)^2/27$ hosts. In reality, however, m must be a natural number, and hence the statement above holds only when $r \bmod 3 = 1$. When $r \bmod 3 = 2$, we can set $m=(r+1)^2/9$ or $m=(r+4)^2/9$, and consequently n becomes $(r+1)^2(r+4)/27$ and $(r+4)^2(r-4)/27$, respectively; since $r>0$, the former case is better. When $r \bmod 3 = 0$, we can set $m=r^2/9$ or $m=(r+3)^2/9$, and consequently n becomes $r^2(r+6)/27$ and $r(r+3)^2/27$, respectively; since $r>0$, the latter case is better. Thus, we can state:

Theorem 6 *An XY-clique host-switch graph can have at most*

$$\begin{cases} \frac{r(r+3)^2}{27}, & \textit{if } r \bmod 3 = 0 \\ \frac{(r+2)^3}{27}, & \textit{if } r \bmod 3 = 1 \\ \frac{(r+1)^2(r+4)}{27}. & \textit{if } r \bmod 3 = 2 \end{cases}$$

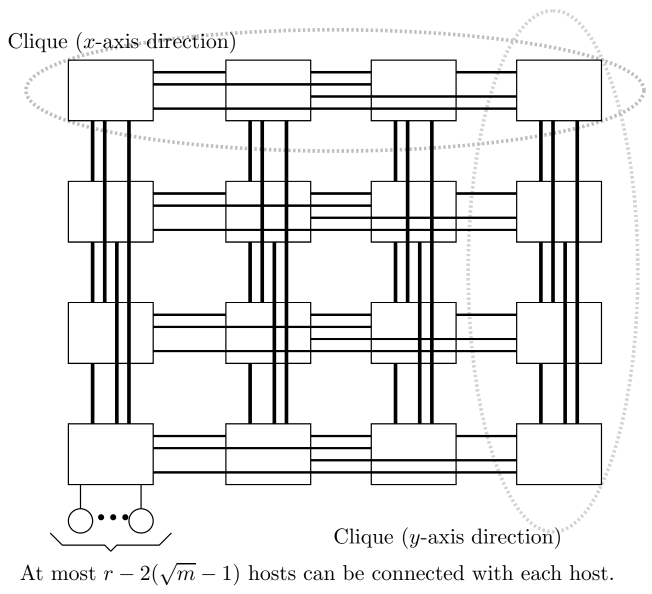

Clique (x-axis direction)
Clique (y-axis direction)
At most $r - 2(\sqrt{m} - 1)$ hosts can be connected with each host.

Fig. 3. An example of an XY-clique host-switch graph.

hosts.

Accordingly, an XY-clique host-switch graph can connect $\Theta(r^3)$ hosts. This is asymptotically better than a star host-switch graph, which can connect $\Theta(r^2)$ hosts. More accurately, we can state:

Theorem 7 *An XY-clique host-switch graph can connect more hosts than a star host-switch graph if and only if* $r \geqslant 19$.

A specific example of an XY-clique host-switch graph is found in the field of computer architecture. It is called the *flattened butterfly* [23].

3.3.3 Polarity Host-Switch Graph

A *polarity graph* [24] (a.k.a. *Brown's construction* or *Brown graph* [25]) is a well-known undirected graph of diameter 2. For a prime power q, the polarity graph $B(q)$ provides an undirected graph with $q^2 + q + 1$ vertices. The maximum degree is $q + 1$; in detail, q^2 vertices have degree $q + 1$, and $q + 1$ vertices have degree q.

We can construct a host-switch graph by connecting hosts to a polarity graph. Let us call such a graph a *polarity host-switch graph*. Clearly we can state:

Lemma 5 *A polarity host-switch graph can connect at most* $r(q^2 + q + 1) - q(q + 1)^2$ *hosts.*

Let $n_{\max}$ be $r(q^2+q+1)-q(q+1)^2$. Since $\partial n_{\max}/\partial q$ is equal to $-3q^2 + 2q(r - 2) + r - 1$, the value of $n_{\max}$ is maximized when $q = (\sqrt{r^2 - r + 1}+r-2)/3$ and $r = (3q^2+4q+1)/(2q+1)$. Substituting this value of r to $r(q^2+q+1)-q(q+1)^2$, $n_{\max}$ becomes $(q^4 + 2q^3 + 4q^2 + 4q + 1)/(2q + 1)$. Thus, a polarity host-switch graph can potentially connect more hosts than a star host-switch graph and an XY-clique host-switch graph.

3.4 Relationship between RDP and h-ASPL

Until now we discuss the RDP, but an important question remains: *does the best host-switch graph in terms of the RDP have the lowest h-ASPL?* Let us consider this question.

When $D(G)$ is equal to 2, the h-ASPL of a host-switch graph with the largest possible hosts (i.e., r hosts) is obviously 2 (i.e., minimum). Thus, the answer to the question above is yes. When $D(G)$ is equal to 3, we can prove that a clique host-switch graph, which can connect the largest possible hosts, takes the minimum value of the h-ASPL when $n > r$ (See Appendix in [26]). Thus, the answer to the question above is also yes.

However, once $D(G)$ exceeds three, there exist no trivial solutions for the RDP. We should thus consider an alternative question: *does* better *host-switch graph in terms of the RDP have* lower *h-ASPL?* We can find counter-evidence to this question. Let us consider the case of biclique host-switch graphs. As we mentioned before, the best biclique host-switch graph in terms of the RDP is a star host-switch graph. However, the h-ASPL of a star host-switch graph in Fig 2b (≈ 3.68) is higher than the h-ASPL of a biclique host-switch graph in Fig 2a (≈ 3.26). In this way, a host-switch graph with larger hosts does not always provide lower h-ASPL.

In summary, the deterministic approach is effective for host-switch graphs of diameter 3 or less, but not effective for those of diameter 4 or more. Thus it would not be appropriate for designing practical interconnection networks. However, we cannot rule out the possibility that a practical host-switch graph can deterministically be constructed, and hence the RDP remains of interest.

4 Heuristic Construction of Host-Switch Graphs

In this section we propose a heuristic approach for constructing a host-switch graph with low h-ASPL. In particular, we present a randomized algorithm with fixed parameters n, m, and r, and discuss the optimal number m switches.

4.1 Overview

First, let us formulate a problem to solve. Unlike the deterministic approach, a heuristic one enables us to optimize h-ASPL directly, and hence the problem is below.

Problem 2 (Order/Radix Problem) *Given natural numbers n and r, find a host-switch graph with order n and radix r that provides the minimum possible value of h-ASPL.*

To solve this problem, we use a randomized algorithm similar to prior studies [27], [28] that searches an undirected regular graph with low diameter/ASPL. We adopt simulated annealing (SA) to escape from a local solution.

4.2 Minimizing h-ASPL

4.2.1 Computation of h-ASPL

Let us begin with a simple case, computing the ASPL of an ordinary undirected graph—not a host-switch graph. To compute the ASPL, we should solve the *all-pairs-shortest paths (APSP)* problem for an undirected unweighted graph.

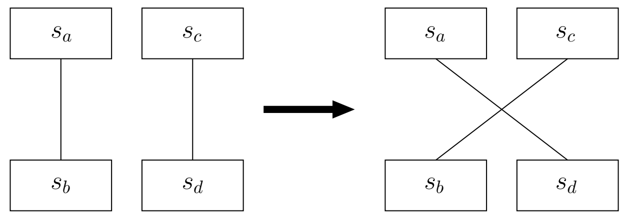


Fig. 4. Swap operation which changes endpoints of two switch-switch edges.

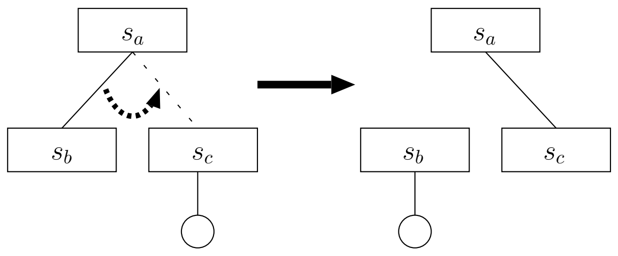


Fig. 5. Swing operation which changes endpoints of a switch-switch edge and a host-switch one.

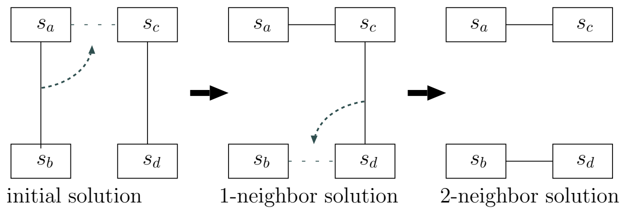


Fig. 6. 2-neighbor swing operation (hosts are omitted for simplicity).

Several algorithms are proposed for the APSP problem. Among them the simplest one is using the breadth-first search (BFS) from every vertex with an $\mathcal{O}(|V||E|)$ running time in total. The h-ASPL is given by $\sum_{0\leqslant i<j<m}(\ell(s_i,s_j)+2)\cdot w_iw_j/\binom{n}{2}$, where w_i denotes the number of hosts connected with s_i. Thus, the time complexity is equal to that of computing the ASPL between switches. As a result, BFS enables us to compute the h-ASPL with an $\mathcal{O}(m^2r-mn)$ running time. Note that the order n does not increase the time complexity. On the contrary, n decreases it. Also, it reduces to $\mathcal{O}(m^2)$ if n takes the upper bound given by Lemma 1, i.e., $n=mr-2(m-1)$.

In our experiments in this paper, we use a BFS-based algorithm because it is fast enough especially for sparse graphs, but we would be able to improve time complexity by using faster algorithms [29] for large m or dense graphs.

4.2.2 Swap Operation: Local Search Restricted to Regular Host-Switch Graphs

Let a *k-regular host-switch graph* (or simply *regular host-switch graph*) $G=(H,S,E)$ denote a host-switch graph such that any switch in S has the fixed number k of neighbor switches and $p-k$ hosts, respectively. We shall begin with a simple algorithm that can be applied only for a regular host-switch graph.

We can use a local search algorithm where a neighbor solution is given by a *swap operation* (Fig. 4), which converts $\{s_a,s_b\},\{s_c,s_d\}\in E$ to $\{s_a,s_d\},\{s_b,s_c\}$. Since the number of hosts per switch of a regular host-switch graph is averagely n/m, the lower bound on the h-ASPL of a k-regular host-switch graph is obtained by the Moore bound [1], the known lower bound on the ASPL of a K-regular graph with N vertices, say $M(N,K)$, as follows:

$$A(G)\geqslant\frac{(n/m)^2\binom{m}{2}(M(m,r-n/m)+2)+2\binom{n/m}{2}m}{\binom{n}{2}}$$
$$=\frac{M(m,r-n/m)(mn-n)}{mn-m}+2. \quad (3)$$

The results of the algorithm using the swap operation is compared with the extended algorithm below.

4.2.3 Swing Operation: Local Search for Any Host-Switch Graph

We extend the algorithm above so that it can change endpoints of host-switch edges as well as those of switch-switch edges. The extended algorithm is based on a new operation called a *swing operation* (Fig. 5). The swing operation converts $\{s_a,s_b\},\{s_c,h_i\}\in E$ to $\{s_a,s_c\},\{s_b,h_i\}$, and hence it changes endpoints of host-switch edges. In other words, this operation changes the number of hosts connected with each switch. Let $\textsc{Swing}(s_a,s_b,s_c)$ denote the swing operation.

As stated above, the swap operation never changes endpoints of host-switch edges, and contrariwise the swing operation always changes them. Thus we should combine them to obtain good solutions. To this end, we introduce a *2-neighbor swing operation* (Fig. 6). Note that hosts are omitted in the figure for simplicity.

The 2-neighbor swing operation has the following four steps:
Step 1: Operate $\textsc{Swing}(s_a,s_b,s_c)$ and evaluate the solution, called the *1-neighbor solution*.
Step 2: If the 1-neighbor solution is accepted, then move to the 1-neighbor solution and the operation ends. Otherwise, go to the next step.
Step 3: Operate $\textsc{Swing}(s_d,s_c,s_b)$ and evaluate the solution, called the *2-neighbor solution*.
Step 4: If the 2-neighbor solution is accepted, then move to the 2-neighbor solution. Otherwise, the initial solution holds.
Consequently, this operation contains both of the swap operation (if the 2-neighbor solution is accepted) and the swing operation (if the 1-neighbor solution is accepted).

4.2.4 Discussion about Optimal Number of Switches

We discuss the optimal number of switches, because a randomized algorithm must fix it during optimization. To discuss it, we carry out SAs with the swap operation and the 2-neighbor swing operation, and compare their results with the lower bound given by Theorem 2 and the Moore bound. In SAs, initial host-switch graphs are constructed randomly. We obtain the results for $n=128,256,512,1024$ and $r=12,24$, and show typical results among them in Fig. 7. We pick up the typical results, and other cases are similar to either of the three results. Here, the Moore bound is calculated by (3); however, n/m must be an integer, and

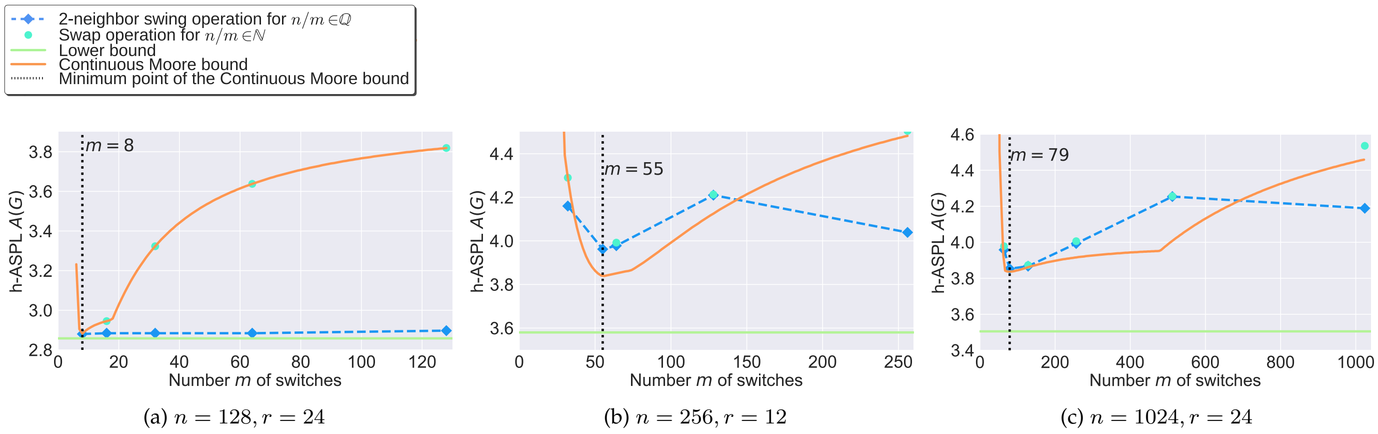


Fig. 7. Relationship between h-ASPL and the number of switches.

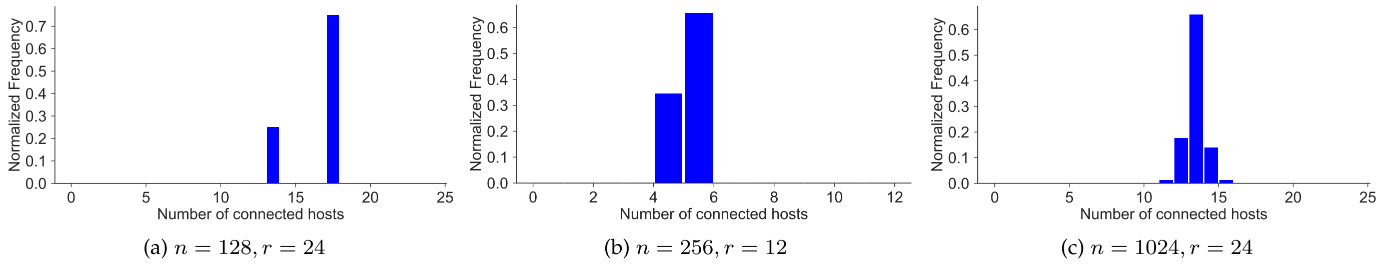


Fig. 8. Host distribution when $m = m_{\text{opt}}$.

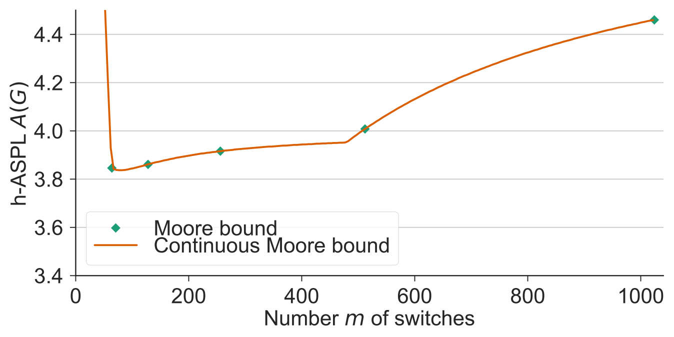


Fig. 9. Comparison between the Moore bound and the continuous Moore bound.

thus the Moore bound of a host-switch graph has a value for specific pairs of n, m, and r. We hence extend the Moore bound so that the degree can be a rational number, not only an integer. We call it the *continuous Moore bound*. In Fig. 9 we show the difference between the Moore bound and the continuous Moore bound in the case of $n = 1024$ and $r = 24$.

The h-ASPL is less than 3 only in the case of $n = 128$ and $r = 24$ (Fig. 7a), because the switches can constitute a clique only in this case as described in Section 3. In other words, n can be less than $m(r - m + 1)$ only in this case (see Lemma 2). Also, in this case, the h-ASPL is close to the lower bound derived by Theorem 2, which suggests it is almost optimal. In the cases of other pairs of n and r, however, $n \gg m(r - m + 1)$ holds for any m, and consequently the h-ASPLs exceed 3.

In Fig. 7, a dotted line represents the number m of switches such that the continuous Moore bound takes the minimum value. The important thing is that this value of m accords with the value of m such that the h-ASPL takes the minimum value in all the cases. Let m_{opt} and A_{opt} denote this value of m and the h-ASPL when $m = m_{\text{opt}}$, respectively. In Fig. 8 we show the distribution of the number of connected hosts of a switch, which we call the *host distribution*. Interestingly, the obtained graph includes switches that have different number of hosts. This corresponds to neither conventional direct nor indirect networks.

Other phenomenon of interest is that, when $m < m_{\text{opt}}$ or $m \gg m_{\text{opt}}$, the h-ASPL of a regular host-switch graph significantly exceeds A_{opt} as compared with the h-ASPL of a non-regular host-switch graph. Let us discuss each case.

Case 1: $m \gg m_{\text{opt}}$.
In this case, there exist unused switches that are not included in any shortest path between hosts, in a non-regular host-switch graph. In the case of $(n, m, r) = (1024, 1024, 24)$ shown in Fig. 7c, the host distribution is similar to Fig. 10, which illustrates over 70% switches connect no hosts. This is similar to indirect networks, but there exists a clear difference. In the case of indirect networks, all the switches are on some shortest path. In our case of $(n, m, r) = (1024, 1024, 24)$, however, many switches are not on any shortest path between hosts, i.e., they are redundant. A regular host-switch graph cannot contain such redundant switches and all the switches must connect hosts.

Case 2: $m < m_{\text{opt}}$
In this case, only a host-switch graph with small number of switches can be constructed. Hence, when a host-switch

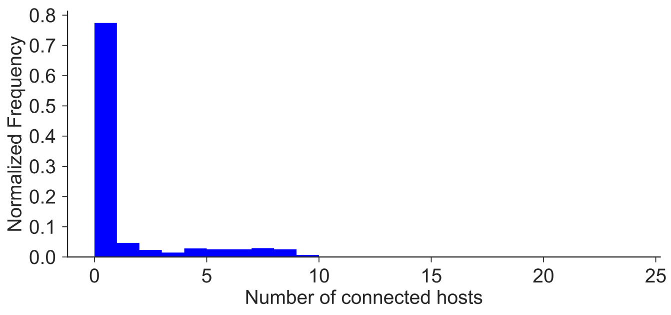


Fig. 10. Host distribution of a host-switch graph with unused switches when $(n, m, r) = (1024, 1024, 24)$.

graph is regular, the degree becomes too small and consequently the h-ASPL drastically increases. When a host-switch graph is not regular, however, a tree-like graph in which only a few switches exist can be constructed. That is why the h-ASPL can be less than the continuous Moore bound.

From the above, we have the essential observation about the optimal number of switches, as follows:

Observation 1 *For fixed n and r, a host-switch graph attains the minimum h-ASPL when it has m switches such that the continuous Moore bound takes the minimum value.*

On the basis of this observation, we carry out the randomized algorithm with fixed m.

4.3 Maximizing BiW

Our randomized algorithm can be applied for optimizing a parameter other than the h-ASPL. Now we focus on the bisection width (BiW) because it is another important metrics for interconnection networks.

4.3.1 Computation of BiW of Host-Switch Graphs

A *bisection width (BiW)* of an ordinary undirected graph G is the minimum number of edges that have to be removed from G to partition it in two halves. In general, interconnection networks with larger BiW are better in terms of performance because minimum cut determines maximum possible flows through a network, according to the max-flow min-cut theorem [30]. Also, the BiW corresponds to the bisection bandwidth of a network if all the links have a fixed bandwidth.

Unlike the h-ASPL, the BiW is hard to compute. Although the BiWs of some specific graphs [31]–[33] and its bounds based on spectral graph theory [34] are studied, its calculation for arbitrary graph is NP-complete [35]. We hence compute it approximately by using a graph partitioning software called hMETIS [36], which is a family of METIS [37] generalized for hypergraphs and provides more accurate results.

Based on the above, we also define the BiW for host-switch graphs. A *bisection width (BiW)* of a host-switch graph is defined as the minimum number of cut edges between two subgraphs with $\lfloor n/2 \rfloor$ and $\lceil n/2 \rceil$ hosts, respectively.[1] A host-switch graph is said to be *full-bisection* if its BiW is more than $\lfloor n/2 \rfloor$, and *half-bisection* if its BiW is between $\lfloor n/4 \rfloor$ and $\lfloor n/2 \rfloor$.

1. Note that this definition is more practical than that in [26], which is the minimum number of cut edges between two subgraphs that include $\lfloor (n+m)/2 \rfloor$ and $\lceil (n+m)/2 \rceil$ vertices, respectively.

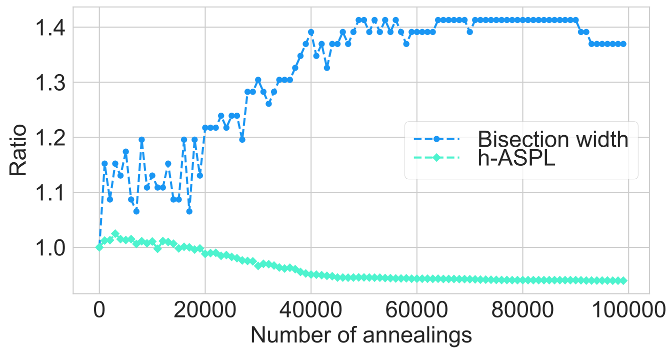


Fig. 11. Changes of h-ASPL and BiW during optimization.

4.3.2 First Attempt: Direct Optimization

We firstly attempt to maximize the BiW directly. To this end, we should change the objective function of SA to the BiW and retune SA's parameters such as the initial temperature. However, this methodology cannot optimize the BiW. There are two reasons as follows. First, an objective function is approximated, and thus SA cannot accurately select a better neighbor solution. Second, the change of the BiW is significant and almost uniform, because it is an integer in contrast to h-ASPL, which is a rational number. For these reasons the BiW should not be included in an objective function.

4.3.3 Reducing h-ASPL Yields Increasing BiW

Our idea to increase the BiW is to optimize another metric correlated with the BiW. Previous studies find some parameters such as maximal congestion [38] are correlated with the BiW, but they are also hard to compute. Hence we hypothesize that h-ASPL is correlated with bisection width and verify it; intuitively, a host-switch graph with low h-ASPL has many paths between any pair of switches on average.

In Fig. 11 we show the changes of the h-ASPL and the BiW when we reduce h-ASPL by SA. Intriguingly the BiW increases as the h-ASPL decreases, though the BiW is not considered in the optimization. Furthermore, the change ratio of the BiW is larger than that of the h-ASPL. We repeatedly run the simulation with various parameters, and similar results are obtained. Based on the results, we have the following observation:

Observation 2 *In a host-switch graph with fixed n, m, and r, reducing the h-ASPL yields increasing the BiW.*

Thus our method described in Section 4.2 can be used also for maximizing the BiW.

4.3.4 Discussion about Relationship between Number of Switches and BiW

We discuss the relationship between the number of switches and the BiW. The same as before, we carry out SAs with the 2-neighbor swing operation.

In Fig. 12 we show typical results. We observe that, except for some plots, the BiW linearly increases as the number of switches increases. We find the exception is provided by indirect networks, and hence we plot direct and indirect networks separately. The exception provides worse BiW. This indicates that direct networks provide better tradeoffs between the h-ASPL and the BiW than indirect networks provide, in the case that we optimize the h-ASPL by SA. The fat-tree gives a better BiW at the cost of higher h-ASPL.

From the results, we have the following observation:

Observation 3 *In a direct host-switch graph with fixed n and r and the minimum h-ASPL, the BiW linearly increases as the number m of switches increases.*

By this observation, we can approximate the minimal number of switches that provides half- or full-bisection host-switch graphs, as with the approximation of the optimal number of switches in terms of the h-ASPL.

4.4 Three Proposed Topologies

So far we have described heuristic approach to find host-switch graphs with low h-ASPL and high BiW. In summary, the heuristic approach is more practical than the deterministic one for the following reasons.

- We can directly optimize the h-ASPL without depending on the diameter (we can also, if necessary, consider both of the h-ASPL and the diameter as with [27]).
- We can generate host-switch graphs with prescribed parameters n, m, and r. In particular, m is essential for host-switch graphs because it strongly affects topological properties.
- We can use various objective functions. However, we find that, in the case of optimizing the BiW, reducing h-ASPL yields increasing the BiW. Interestingly, the BiW of host-switch graph with the minimized h-ASPL linearly increases as the number of switches up to certain number in the cases of direct networks.

Notwithstanding, the deterministic approach based on the RDP is still of great interest from a theoretical viewpoint.

On the basis of the results in this section, we propose three topologies, as follows:

1) **Minimum h-ASPL**: a host-switch graph with m_{opt}.
2) **Full-bisection**: a full-bisection host-switch graph with minimum m.
3) **Half-bisection**: a half-bisection host-switch graph with minimum m.

Designers can select them depending on technical requirements. The three topologies are evaluated in Section 5 with existing topologies.

5 Evaluation

Hitherto we might neglect practical viewpoints. However, this section can compensate it; we evaluate our proposed topologies with other topologies, including topologies proposed in Section 3, previously proposed topologies, and existing topologies applied to supercomputers ranked in TOP500 and demonstrate the advantage of our proposed topologies.

5.1 Previously proposed topologies

5.1.1 WK-recursive networks

The WK-recursive network [39] $\mathrm{WK}(K, L)$ is recursively defined with two parameters, the degree K and the level L, as follows.

1) $\mathrm{WK}(K, 1)$ is a K-clique.
2) $\mathrm{WK}(K, L)$ for $L > 1$ is a K-clique regarding $\mathrm{WK}(K, L-1)$ as a node (called a *virtual node*). Here, any two virtual nodes are connected with exactly one edge, and the degree of the nodes must be equal to or less than K.

Regarding a node of a WK-recursive network as a switch, we can construct a host-switch graph. From the definition above, the number m of switches of $\mathrm{WK}(K, L)$ is

$$m = K^L. \tag{4}$$

Each switch is connected to at least $K-1$ other switches, and the number of switch-switch links is incremented by $K-1$ with each recursive call. Thus, the following is satisfied:

$$r \geqslant K, \tag{5}$$

$$n \leqslant K^L(r - K + 1) - (L-1)(K-1). \tag{6}$$

5.1.2 Recursive dual-net

The recursive dual-net $\mathrm{RDP}^k(B)$ [40] is an interconnection network based on a recursive dual-construction of a base network B where k is a level of the recursion. According to [40], a k-level dual-construction for $k > 0$ creates a network containing $(2m_B)^{2^k}/2$ nodes and $d_B + k$ links per node, where m_B and d_B denote the number of nodes and the number of links per node of a base network B, respectively.

Thus, a host-switch graph based on a recursive dual-net satisfies:

$$m = (2m_B)^{2^k}/2, \tag{7}$$

$$n \leqslant (r - k - d_B) \cdot (2m_B)^{2^k}/2, \tag{8}$$

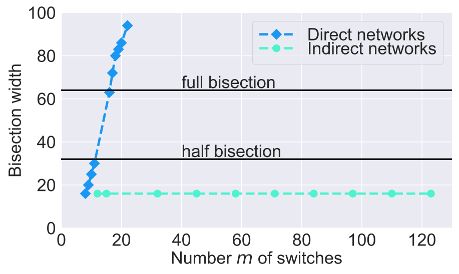
$$r > d_B + k. \tag{9}$$

5.2 Existing topologies

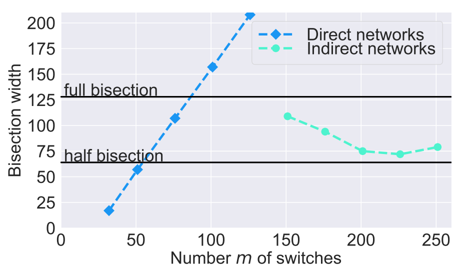
There are many existing topologies for practical interconnection networks. Among them, we pick up three typical topologies: the torus [41], the dragonfly [15], and the fat-tree [42]. They are applied to supercomputers ranked in TOP500 for June 2017 [43]; for example, Titan [44] and Sequoia [45] use the torus, Cori [16] and Piz Daint [17] use the dragonfly, and Tianhe-2 [20] uses the fat-tree. We review them as a host-switch graph to compare them with our proposed host-switch graphs. Note that the definitions described below are specialized for comparisons and there can be other variants of each topology.

5.2.1 Torus

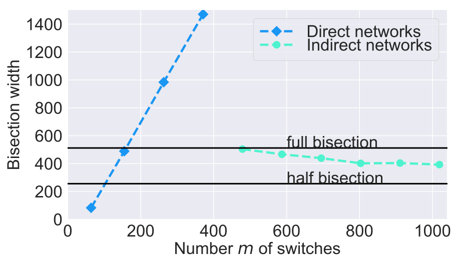
A *K-ary N-torus host-switch graph* is a host-switch graph with additional parameters K and N. Each switch is identified by a N-bit base-K address, $a_{N-1}a_{N-2}\cdots a_0$, and connected to switches with addresses $a'_{N-1}a'_{N-2}\cdots a'_0$ where $a'_i \pm 1 \pmod K = a_i$ for any i $(0 \leqslant i \leqslant N-1)$ and $a'_j = a_j$ for all j $(0 \leqslant j \leqslant N-1$ and $j \neq i)$.



(a) $n = 128, r = 24$



(b) $n = 256, r = 12$



(c) $n = 1024, r = 24$

Fig. 12. Relationship between the BiW and the number of switches.

From the above, the number m of switches of a K-ary N-torus host-switch graph is

$$m = K^N. \tag{10}$$

Since each switch is connected with $2N$ other switches, the order n and the radix r of a torus host-switch graph satisfy:

$$n \leqslant (r - 2N) \cdot K^N, \tag{11}$$
$$r > 2N. \tag{12}$$

5.2.2 Dragonfly

A *dragonfly host-switch graph* is a host-switch graph with additional parameters a, h, g, and p. The switches are divided into g groups, each of which has a switches that construct a clique. Each switch is connected with p hosts and h switches in other groups so that the groups constitute a clique if we regard a group as a node; consequently, g must be equal to $ah + 1$. According to the original paper [15], the parameters should satisfy $a = 2h = 2p$ to balance traffic loads, and hence we assume this equation holds.

From the above, the number m of switches of a dragonfly host-switch graph is

$$m = ag = \frac{a^3}{2} + a. \tag{13}$$

The radix r of a dragonfly host-switch graph is

$$r = (a - 1) + h + p = 2a - 1. \tag{14}$$

The order n of a dragonfly host-switch graph satisfies:

$$n \leqslant mp = \frac{a^4}{4} + \frac{a^2}{2}. \tag{15}$$

5.2.3 Fat-tree

There exist many variants of fat-trees. In this paper, we adopt a three-layer fat-tree such that the number of ports of a switch is uniform, which is a special instance of Clos network called a *K-ary fat-tree* [42]. It is an indirect network unlike the torus and the dragonfly.

A *K-ary fat-tree host-switch graph* is a host-switch graph that corresponds to a K-ary fat-tree consisting of three layers: the core layer with $K^2/4$ switches, the aggregation layer with $K^2/2$ switches, and the edge layer with $K^2/2$ switches. Thus the number m of switches of a K-ary fat-tree host-switch graph is

$$m = 5K^2/4. \tag{16}$$

Each switch in the edge layer can be connected with $K/2$ hosts. Thus the order n of a K-ary fat-tree host-switch graph satisfies:

$$n \leqslant K^3/4. \tag{17}$$

The value of K corresponds to the number of links for each switch, and thus the radix r is simply

$$r = K. \tag{18}$$

5.3 Fundamental Properties

Table 1 summarizes the fundamental properties of host-switch graphs presented in Sections 3, 4, 5.1, and 5.2. We have considered graphs presented in Sections 3 and 4, so let us now compare them with the graphs presented in Section 5.1 and 5.2. The WK-recursive network is identical to a clique host-switch graph when $L = 1$. However, once L becomes greater than 1, the diameter rapidly increases with $\mathcal{O}(2^L)$. When $L = 2$, the diameter is the same as that of the dragonfly. When $L = 3$, the diameter becomes nine, which is excessively large. Thus, let us compare WK$(K, 2)$ and the dragonfly.

One might notice that WK$(K, 2)$ and the dragonfly are quite similar; technically, WK$(K, 2)$ is a dragonfly such that a is equal to g, h is equal to 0 or 1, and p is variable (cf. Section 5.2.2). Let us consider WK$(K, 2)$ with the parameters of the dragonfly for comparison. Since K corresponds to a, the number m of switches is

$$m = a^2. \tag{19}$$

The order n must satisfy:

$$n \leqslant a^2(r - a + 1) - (a - 1) = -a^3 + a^2(r + 1) - a + 1. \tag{20}$$

When we assume $r = 2a - 1$ as with the dragonfly, (20) reduces to

$$n \leqslant a^3 - a + 1. \tag{21}$$

As a result, the dragonfly can connect more hosts than WK$(K, 2)$ when $a > 3$.

Next, we consider the recursive dual-net. We may say that its diameter is large; even if $k = 1$ and $D(B) = 1$ (i.e., B is a clique), the diameter is 6, which is the same as the diameter of the fat-tree. Let us compare the recursive dual-net with $k = 1$ and the fat-tree in terms of scalability. When $D(B) = 1$, d_B must be $m_B - 1$, and k is equal to one. Thus, the order n is not greater than $(r - m_B) \cdot 2m_B^2$ and consequently grows

TABLE 1
Summary of fundamental properties of host-switch graphs.

Graph	**Section**	**Parameters**	**Order** n	**#Switches** m	**Radix** r	**Diameter**
Clique	3	–	$m(r-m+1)$	$\leqslant r+1$	given	3
Star	3	–	$r(r-1)$	$r+1$	given	4
XY-Clique	3	–	$m(r-2\sqrt{m}+2)$	$< (r+2)^2/4$	given	4
Polarity	3	q	$r(q^2+q+1)-q(q+1)^2$	q^2+q+1	$> q+1$	4
Randomly-optimized	4	–	$(r-1)^{D(G)-1}+1$	variable	given	$\geqslant \lceil \log_{r-1}(n-1) \rceil + 1$
WK-recursive	5.1	K, L	$K^L(r-K+1)-(L-1)(K-1)$	K^L	$\geqslant K$	2^L+1
Recursive dual-net	5.1	k, m_B, d_B	$(r-k-d_B)\cdot(2m_B)^{2^k}/2$	$(2m_B)^{2^k}/2$	$> d_B+k$	$2^k D(B)+2^{k+1}$
Torus	5.2	K, N	$(r-2N)\cdot K^N$	K^N	$> 2N$	$\lfloor K/2 \rfloor \cdot N+2$
Dragonfly	5.2	a	$a^4/4+a^2/2$	$a^3/2+a$	$2a-1$	5
Fat-tree	5.2	K	$K^3/4$	$5K^2/4$	K	6

with $\mathscr{O}(mr - m\sqrt{m})$. On the other hand, the order of the fat-tree is clearly $\mathscr{O}(mr)$. Thus, the fat-tree is more scalable than the recursive dual-net.

We experimentally compare our proposed topology with the torus, the dragonfly, and the fat-tree because they are used in real systems, and the dragonfly and the fat-tree are more scalable than the WK-recursive network and the recursive dual-net as we have shown above.

5.4 Experimental Method

The existing topologies above and our three topologies are compared in terms of performance, topological properties (the h-ASPL and the BiW), power consumption, and cost breakdowns. Since each existing topology must take a specific combination of n, m, and r, we separately compare each topology with our topologies. Note that our proposed topologies can construct for any combination of n, m, and r. The comparisons include two experiments below.

5.4.1 Evaluation of Performance and Topological Properties

The performance is evaluated by SimGrid discrete event simulator (v3.15) [46]. One of the APIs implemented in SimGrid, called SMPI, can simulate unmodified MPI applications. We use a shortest path routing scheme using the Floyd-Warshall algorithm. Each host has a computation speed of 100 GFLOPS in all the networks. We configure SimGrid to use its built-in version of the MVAPICH2 implementation of MPI collective communications. For each topology, we generate a SimGrid platform called Autonomous System (AS) [47] and simulate MPI implementation of NAS parallel benchmarks (v3.3.1, Class A for IS and FT and Class B for the others) [48].

Since the NAS parallel benchmarks work only when the number of processes is the power of four, we assume n is equal to 1024 and set the network size to connect 1024 hosts. Each existing topology is constructed by the smallest host-switch graph such that the number of connectable hosts is at least 1024, and 1024 hosts are sequentially connected to switches. Our topologies are constructed so that r becomes the same as each existing topology. Afterwards, hosts are sequentially connected to switches in depth-first order by backtracking.

Table 2 summarizes the parameters and the topological properties of nine topologies used in the experiments. We adopt the torus such that the dimension N is 5 (i.e., 5-D torus), which is used in Sequia. From (10)–(12) we set K and r to 3 and 15, respectively. Consequently the torus satisfies $n \leqslant 1215$, $m = 243$, and $r = 15$. From (13)–(15) we set a to 8 for the dragonfly, and consequently it satisfies $n \leqslant 1056$, $m = 264$ and $r = 15$. Since both the torus and the dragonfly require 15-port switches, our topologies are constructed with 15-port switches to compare with them. From (16)–(18) we adopt 16-ary fat-tree, and consequently the fat-tree satisfies $n \leqslant 1024$, $m = 320$, and $r = 16$. To compare with the fat-tree, our topologies are constructed with 16-port switches.

From Table 2, we notice that the continuous Moore bound for our proposed topologies is 13–15% larger than the lower bound derived by Theorem 2. These differences are mainly caused by the assumed host distribution and the assumed value of m are different. While the continuous Moore bound assumes the host is regularly connected and m is equal to m_{opt}, the lower bound derived by Theorem 2 holds for any host-distribution and any value of m, which is more general and less tight.

5.4.2 Evaluation of Power Consumption and Cost Breakdown

The power consumption and cost breakdowns are evaluated on the basis of models of Mellanox InfiniBand FDR10 switches and Mellanox InfiniBand FDR10 40Gb/s QSFP cables [2]. A physical floorplan is designed so that it is large enough to align all the cabinets on a 2-D grid. Each cabinet is 60 cm wide and 210 cm deep including space for the aisle, and the number of cables and their lengths are calculated. If a cable length is over 100cm, the cable is assumed to be an electrical cable. Otherwise, the cable is assumed to be an optical cable. The network sizes are the same as those in the performance evaluation.

5.5 Results and Discussion

5.5.1 Comparison with 5D Torus

From Table 2, the h-ASPL of the torus (≈ 5.34) is much higher than the continuous Moore bound (≈ 4.47). On the other hand, our topologies have low h-ASPLs close to the Moore bound. Also in terms of the BiW, all of our topologies are better than the torus. It is interesting to note that our topology with the minimum h-ASPL and the half-bisection one provide similar topological properties.

In Fig. 13a we show the results of the performance comparison. Our topology with the minimum h-ASPL outperforms the torus by 22% on average (given by the geometric mean). It achieves particularly high performance in the cases of IS (Integer Sort), FT (Fast Fourier Transform),

TABLE 2
Summary of nine topologies to connect more than 1024 hosts for simulation.

Topology	Radix	# of switches	h-ASPL	Continuous Moore bound	Lower bound by Theorem 2	BiW
5D Torus	15	243	5.34	4.47	3.87	240 (46.9%)
Dragonfly	15	264	4.68	4.48	3.87	272 (53.1%)
Minimum h-ASPL	15	194	4.45	4.45	3.87	297 (58.0%)
Half-bisection	15	184	4.46	4.45	3.87	267 (52.1%)
Full-bisection	15	284	4.51	4.49	3.87	518 (101%)
Fat-tree	16	320	5.86	4.44	3.84	512 (100%)
Minimum h-ASPL	16	183	4.36	4.34	3.84	308 (60.2%)
Half-bisection	16	165	4.36	4.34	3.84	256 (50.0%)
Full-bisection	16	259	4.41	4.38	3.84	515 (101%)

and MG (Multi-Grid), because they require random memory accesses, all-to-all communications, and long-distance communications, respectively, which are not appropriate for regular structure with locality. Our half-bisection topology provides similar performance as that with the minimum h-ASPL. This is because the numbers m of switches and the h-ASPLs of those topologies are similar (see Table 1). Our full-bisection topology provides the best performance. It outperforms the torus by 45% on average.

In Fig. 13b we show the results of the power comparison. Our two topologies, one with the minimum h-ASPL and half-bisection one, consume 20% and 24% lower power as compared with the torus, respectively. This is because the numbers m of switches are smaller than that of the torus. Our full-bisection topology consumes 17% more power as compared with the torus. However, the increasing ratio is less than that of the performance (45%).

In Fig. 13c we show the results of the cost comparison. Here cost breakdowns including switch and cable costs are shown. The results of switch costs are the same as the results of power comparison relatively. The results of cable costs, however, are slightly different; the cable costs of our topologies are larger than those of the torus. This is because our topologies may have long cables to provide low h-ASPLs while the torus requires only short cables. In total, however, the cost of our topologies are not significant.

In Fig. 13d we show the results of the performance per watt. Because of the reduction of the power consumption, two of our topologies drastically improve the performance per watt. In particular, our half-bisection topology provides the best improvement (61% on average). On the other hand, our full-bisection topology improves slightly since it consumes large power.

Overall, as compared with the torus, our three topologies provide higher performance. In addition, two of them consume smaller power consumption and costs. One of them, the full-bisection topology, consumes more power consumption and costs, but it attains the best performance and its improvement ratio is more than the increasing ratio of power consumption and costs. In terms of the performance per watt, our half-bisection topology is the best.

5.5.2 Comparison with Dragonfly

From Table 2, the dragonfly provides good topological properties. Its h-ASPL (≈ 4.68) is close to the continuous Moore bound (≈ 4.48) and its BiW ($\approx 53.1\%$) is more than $n/4$ (i.e., 50%). Hence we can confirm that the dragonfly is near optimal topology with the specific pair of n, m, and r. Notwithstanding, our topologies can slightly reduce h-ASPL of the dragonfly, and two of them reduce the number of switches.

In Fig 14a. we show the results of the performance comparison. Our topology with the minimum h-ASPL outperforms the dragonfly by 12% on average. These results illustrate a different tendency from the comparison with the torus, because the dragonfly provides low h-ASPL and the performance does not degrade even when the long-distance traffic occurs. These results substantiate that the h-ASPL is important metrics for performance. It would be strange that the EP benchmark is performing poorer on the proposed networks than the dragonfly. This is because the EP benchmark requires few communications, and hence the h-ASPL has little effect on the performance. But rathar, the application mapping affects the performance. Note that the performance of the EP is hard to change even if the network changes.

In Fig. 14b we show the results of the power comparison. The results of our three topologies are the same as the case of comparison with the torus since the radix is the same. The dragonfly consumes more power than the torus, and thus our topologies can efficiently reduce the power consumption.

In Fig. 14c we show the results of the cost comparison. Here we assume the switches in a group are located in a rack, and hence cable costs are small as compared with in the case of comparison with the torus. The switch costs consequently occupy a majority of total costs, and our topologies can effectively save costs.

In Fig. 14d we show the results of the performance per watt. Since the dragonfly consumes larger power than the torus does, the improvements of the performance per watt by our topologies becomes more significant. As shown in the figure, our topologies improve the performance per watt of the dragonfly in up to 60%. Interestingly, this ratio is almost the same of the ratio in Fig. 13d.

Overall, as compared with the dragonfly, our three topologies provide higher performance. In addition, two of them consume smaller power consumption and costs. One of them, the full-bisection topology, consumes more power consumption and costs, but the increasing ratio is less than that of performance. Since using racks reduces cable costs, switch costs become significant, and consequently our topologies for comparison with the dragonfly can effectively save costs and reduce power consumption. In terms of the performance per watt, our half-bisection topology is the best.

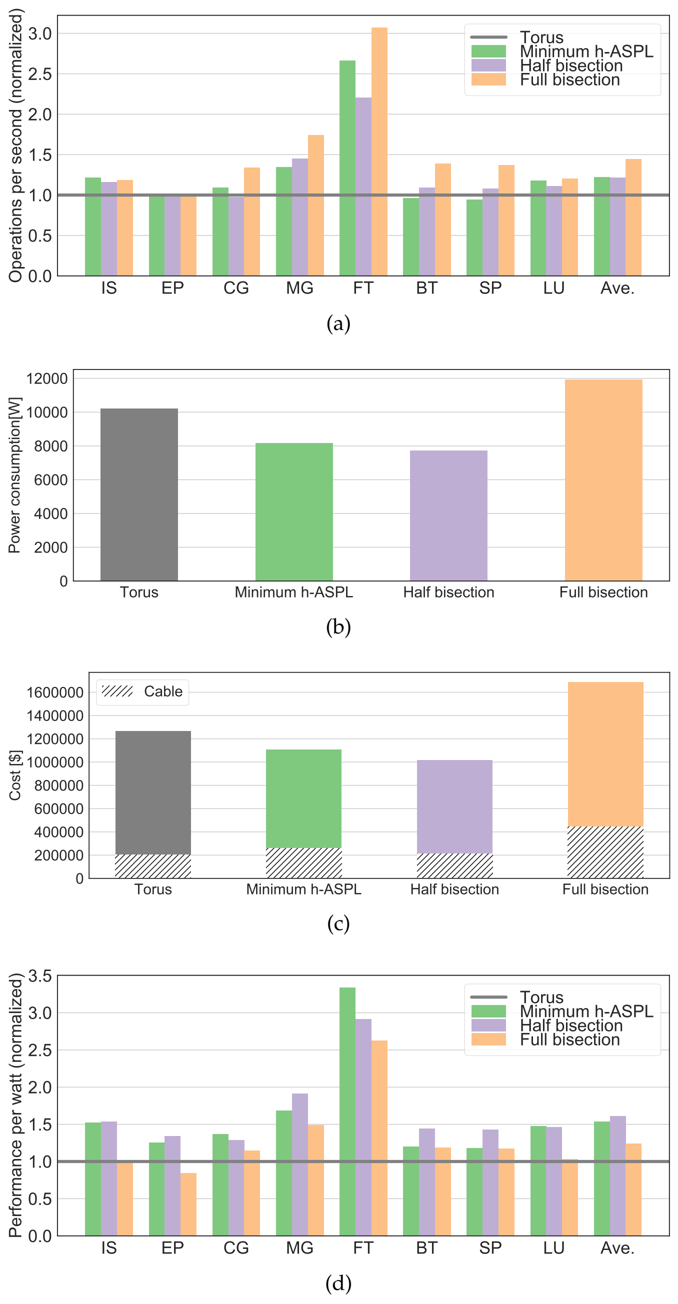


Fig. 13. Results of comparisons between torus and proposed topology: (a) Performance (eight benchmarks and the geometric mean); (b) Power consumption; (c) Cost breakdown (Cable and Switch); (d) Performance per watt.

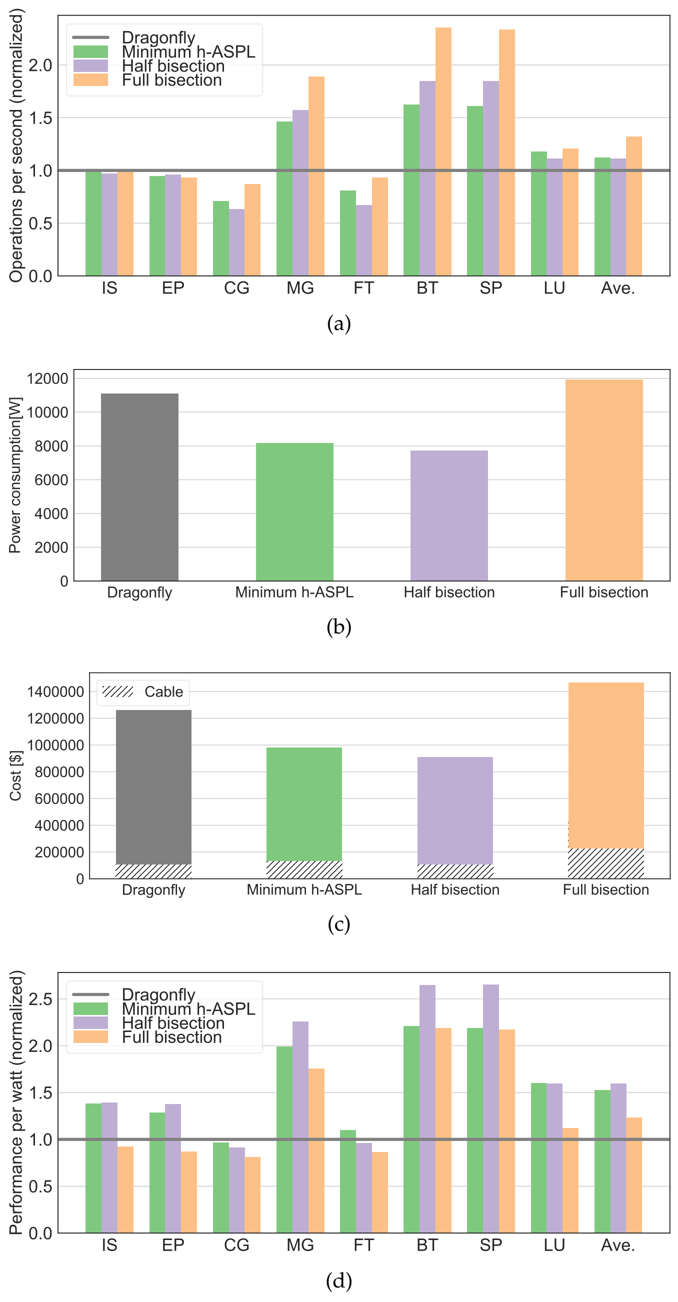


Fig. 14. Results of comparisons between dragonfly and proposed topology: (a) Performance (eight benchmarks and the geometric mean); (b) Power consumption; (c) Cost breakdown (Cable and Switch); (d) Performance per watt.

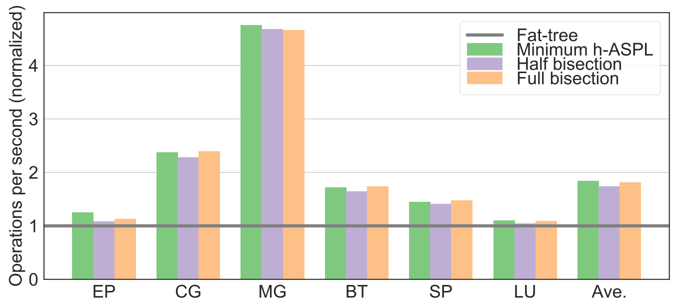
5.5.3 *Comparison with Fat-tree*

From Table 2, the fat-tree has the highest h-ASPL (≈ 5.86), which is much higher than the continuous Moore bound (≈ 4.44). It is full-bisection, but, because of that, the number of switches becomes the most. From these results, we can say the fat-tree is far from optimum in terms of the h-ASPL, the BiW, and switch costs.

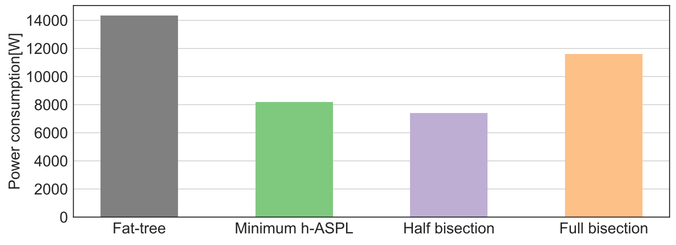
In Fig. 15a we show the results of the performance comparison (due to computational complexity, simulations for IS and FT are omitted). Our topology with the minimum h-ASPL outperforms the fat-tree by 84% on average. The results are similar to that in Fig. 13a , because the fat-tree has also regular structure with locality and the h-ASPL is high. In particular, the fat-tree degrades performance especially in MG, a memory intensive application that requires long-distance communications; all of our topologies are more than 4 times faster than the fat-tree.

In Fig. 15b we show the results of the power comparison. Unlike the torus and the dragonfly, the fat-tree consumes more power than all of our topologies. The power consumption of our topolgoies is almost the same as that of our topologies for comparison with the torus and the dragonfly. This is because the number of switches is reduced while the radix increases.

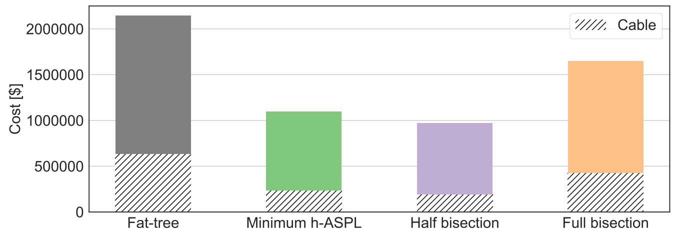
In Fig. 15c we show the results of the cost comparison. Unlike the torus and the dragonfly, the fat-tree requires



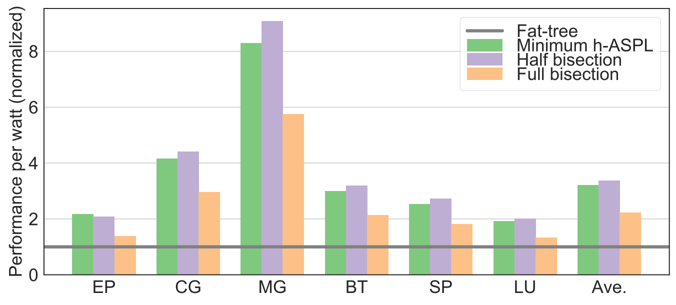
(a)



(b)



(c)



(d)

Fig. 15. Results of comparisons between fat-tree and proposed topology: (a) Performance (six benchmarks and the geometric mean); (b) Power consumption; (c) Cost breakdown (Cable and Switch); (d) Performance per watt.

not only higher cable costs but also higher switch costs as compared with all of our topologies. This is because the number of switches of the fat-tree is large.

In Fig. 15d we show the results of the performance per watt. This shows the most drastic improvement, because our topologies can efficiently improve both the performance and the power consumption. As in the results above (Figs. 13d and 14d), our half-bisection topology provides the best improvement.

Overall, as compared with the fat-tree, our topologies drastically improve performance with lower power consumption and costs. This result indicates that indirect networks are not good solutions for high-performance interconnection networks in terms of the end-to-end latency. In terms of the performance per watt, our half-bisection topology is the best. From the results of the three comparisons thus far, we can say that our half-bisection topology provides the best power efficiency.

5.6 Practical Feasibility and Limitations

Finally, we discuss practical feasibility and limitations of the proposed topologies.

5.6.1 *Dead-lock Free Routing and Routing Tables*

Our proposed topologies require a method for guaranteeing dead-lock freedom since they have no regular structure. Many methods that can be applied to non-structured topologies have been proposed until recently: avoiding cyclic dependencies [49]–[51], using virtual channels [52], and forwarding every flit in the deadlocked ring at the same time [53]. There exist trade-offs between them in terms of the performance (whether the path is the shortest path or not) and the number of virtual channels, and hence we should select the preferred method according to design requirements.

In addition, our proposed topologies require routing tables, and consequently the scale of topologies can be limited by routing table size at each switch. However, we note that most of the supercomputers listed in TOP500 are based on Ethernet or InfiniBand. For all these systems, the routing table size is thus also a scalability limitation regardless of the network topology.

5.6.2 *Application Mapping*

A concern with non-structured topologies is the mapping of application processes to compute nodes. Conventional topologies can match application communication patterns such as lattice communication, and decades of parallel computing research have gone into designing algorithms to compute efficient mappings of classes of applications. The more random and unstructured the topology, the more difficult it is to determine a good application mapping. However, for parallel applications with dynamic workloads and irregular parallel applications with non-deterministic or complex communication patterns, it is difficult to compute an efficient mapping of the processes to the compute nodes even in a structured topology.

5.6.3 *Combinations of n, m, and r*

Our proposed topologies also relax a practical limitation. Conventional topologies can be designed with specific combinations of n, m, and r under strict conditions (as shown in Table 1) although the scale of a system should be determined based on power budget and costs. On the other hand, our proposed topologies can be designed with arbitrary r and variable m. As we have described until now, the value of m is particularly important for the h-ASPL and the BiW, and also for the power consumption and costs.

6 Conclusions

In this paper we have presented a novel graph called a host-switch graph, which consists of two types of vertices, hosts and switches. The degree of each host and each switch is 1 and r, respectively, and thus a host-switch graph represents the topology of a computer network with 1-port host computers and r-port switches. We firstly focus on the host-to-host average shortest path length (h-ASPL) and formulates an optimization problem called the order/radix problem: given order and radix, find a host-switch graph with the minimum objective function. For this problem, we show the lower bound on the h-ASPL and present a randomized algorithm based on the 2-neighbor swing operation. We show the optimal number of switches that provides the minimum h-ASPL can be approximated by the continuous Moore bound.

Furthermore, we empirically show that reducing the h-ASPL yields increasing the BiW as a side-effect. In the case of direct networks, the BiW linearly increases as the number of switches increases; in the case of indirect networks, on the other hand, there exists no obvious relationship between the h-ASPL and the bisection width. We can thus approximate the minimum number of switches for direct networks to provide a certain BiW. Based on the experimental results, we have proposed three topologies, which are given by the host-switch graph with the minimum h-ASPL, half- and full-bisection host-switch graphs, respectively.

We have compared the proposed three topologies with existing topologies applied to supercomputers ranked in TOP500, the torus, the dragonfly, and the fat-tree, in terms of performance, topological properties (the h-ASPL and the BiW), power consumption, and cost breakdowns. Our results demonstrate that, when the number of hosts is 1024, all the proposed topologies outperform existing topologies in terms of operation per second for MPI applications by 11%–84% on average. The topology with the minimum h-ASPL and the half-bisection topology can reduce the number of switches by 20%–48%. As a result, our topologies can effeciently improve the performance per watt; in particular, we have shown that the half-bisection topology is the best in terms of the performance per watt. Thus we have successfully demonstrated that our method can directly be used for designing interconnection networks.

Host-switch graphs arguably help theoreticians to discuss practical interconnection networks without technical knowledge thereof and, at the same time, provide theoretical base for engineers who are not familiar with graph theory. In this paper, we substantiate that direct networks are better than indirect networks in terms of the h-ASPL, the BiW, and switch costs. In particular, we confirm the dragonfly is near optimal, but the number of switches should be further reduced. In this sense, the study of host-switch graphs bridges a gap between graph theory and computer engineering.

References

[1] M. Miller and J. Širáň, "Moore graphs and beyond: A survey of the degree/diameter problem," *Electron. J. Combinatorics*, vol. DS14, pp. 1–61, electronic only, 2005.

[2] M. Besta and T. Hoefler, "Slim fly: A cost effective low-diameter network topology," in *Proc. Int. Conf. High Performance Computing, Networking, Storage and Analysis*, Nov. 2014, pp. 348–359.

[3] R. Mizuno and Y. Ishida, "Constructing large-scale low-latency network from small optimal networks," in *Proc. Int. Symp. Netw.-on-Chip*, Sep. 2016, pp. 1–6.

[4] F. Lei, D. Dong, X. Liao, X. Su, and C. Li, "Galaxyfly: A novel family of flexible-radix low-diameter topologies for large-scales interconnection networks," in *Proc. Int. Conf. Supercomputing*, Jun. 2016, pp. 1–12.

[5] "GraphGolf: the order/degree problem competition," http://research.nii.ac.jp/graphgolf/, 2017.

[6] B. Bollobás and F. R. K.Chung, "The diameter of a cycle plus a random matching," *SIAM J. Discrete Math.*, vol. 1, pp. 328–333, 1988.

[7] P. Erdős and A. Rényi, "On random graphs I," *Publicationes Mathematicae (Debrecen)*, vol. 6, pp. 290–297, 1959.

[8] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, pp. 440–442, 1998.

[9] M. Koibuchi, H. Matsutani, H. Amano, D. F. Hsu, and H. Casanova, "A case for random shortcut topologies for HPC interconnects," in *Proc. Int. Symp. Comput. Archit.*, Jun. 2012, pp. 177–188.

[10] A. Singla, C.-Y. Hong, L. Popa, and P. B. Godfrey, "Jellyfish: Networking data centers randomly," in *Proc. USENIX Conf. Networked Syst. Design and Implementation*, Apr. 2012, pp. 17:1–17:14.

[11] U. Y. Ogras and R. Marculescu, "It's a small world after all: NoC performance optimization via long-range link insertion," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 14, pp. 693–706, Jul. 2006.

[12] M. Koibuchi, I. Fujiwara, H. Matsutani, and H. Casanova, "Layout-conscious random topologies for HPC off-chip interconnects," in *Proc. Int. Symp. High Performance Comput. Archit.*, Feb. 2013, pp. 484–495.

[13] J. Flich, T. Skeie, A. Mejía, O. Lysne, P. López, A. Robles, J. Duato, M. Koibuchi, T. Rokicki, and J. C. Sancho, "A survey and evaluation of topology-agnostic deterministic routing algorithms," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, pp. 405–425, 2012.

[14] C. L. Seitz, "The cosmic cube," *Commun. ACM*, vol. 28, Jan. 1985.

[15] J. Kim, W. J. Dally, S. Scott, and D. Abts, "Technology-driven, highly-scalable dragonfly topology," in *Proc. Int. Symp. Comput. Archit.*, Jun. 2008, pp. 77–88.

[16] K. Antypas, N. Wright, N. Cardo, A. Andrews, and M. Cordery, "Cori: a Cray XC pre-exascale system for NERSC," in *Proc. Cray User Group Conf.*, May 2014.

[17] S. R. Alam, T. Athanassiadou, T. W. Robinson, G. Fourestey, A. Jocksch, L. Marsella, J.-G. Piccinali, J. Poznanovic, B. Cumming, and D. Ulmer, "First 12-cabinets Cray XC30 system at CSCS: Scaling and performance efficiencies of applications," in *Proc. Cray User Group Conf.*, May 2013.

[18] C. E. Leiserson, "Fat-trees: Universal networks for hardware-efficient supercomputing," *IEEE Trans. Comput.*, vol. C-34, pp. 892–901, Oct. 1985.

[19] W. D. Hillis and L. W. Tucker, "The CM-5 connection machine: A scalable supercomputer," *Commun. ACM*, vol. 36, Jan. 1993.

[20] X.-K. Liao, Z.-B. Pang, K.-F. Wang, Y.-T. Lu, M. Xie, J. Xia, D.-Z. Dong, and G. Suo, "High performance interconnect network for Tianhe system," *J. Comput. Sci. Technol.*, vol. 30, Mar. 2015.

[21] B. Elspas, "Topological constraints on interconnection-limited logic," in *Proc. Symp. Switching Circuit Theory Logical Design*, Nov. 1964, pp. 133–137.

[22] L. G. Roberts and B. D. Wessler, "Computer network development to achieve resource sharing," in *Proc. Spring Joint Comput. Conf.*, May 1970, pp. 543–549.

[23] J. Kim, W. J. Dally, and D. Abts, "Flattened butterfly: A cost-efficient topology for high-radix networks," in *Proc. Int. Symp. Comput. Archit.*, Jun. 2007, pp. 126–137.

[24] M. Bachratý and J. Širáň, "Polarity graphs revisited," *Ars Mathematica Contemporanea*, vol. 8, pp. 55–67, 2015.

[25] W. G. Brown, "On graphs that do not contain a Thomsen graph," *Canad. Math. Bull*, pp. 281–285, Feb. 1966.

[26] R. Yasudo, M. Koibuchi, K. Nakano, H. Matsutani, and H. Amano, "Order/radix problem: Towards low end-to-end latency interconnection networks," in *Proc. Int. Conf. Parallel Process.*, Aug. 2017, pp. 322–331.

[27] K. Nakano, D. Takafuji, S. Fujita, H. Matsutani, I. Fujiwara, and M. Koibuchi, "Randomly optimized grid graph for low-latency interconnection networks," in *Proc. Int. Conf. Parallel Process.*, Aug. 2016, pp. 340–349.

[28] N. Shimizu and R. Mori, "Average shortest path length of graphs of diameter 3," in *Proc. Int. Symp. Netw.-on-Chip*, Sept. 2016, pp. 1–6.

[29] T. M. Chan, "All-pairs shortest paths for unwrighted undirected graphs in o(mn) time," in *Proc. ACM-SIAM Symp. Discrete Algorithm*, Jan. 2006, pp. 514–523.

[30] P. Elias, A. Feinstein, and C. E. Shannon, "A note on maximum flow through a network," *IRE Trans. Inform. Theory*, vol. 2, pp. 117–119, Dec. 1956.

[31] K. Nakano, "Linear layouts of generalized hypercubes," *Int. J. Foundations Comput. Sci.*, vol. 14, pp. 137–156, Feb. 2003.

[32] J. Díaz, M. J. Serna, and N. C. Wormald, "Bounds on the bisection width for random d-regular graphs," *Theoretical Comput. Sci.*, vol. 382, pp. 120–130, Aug. 2007.

[33] J. A. Aroca and A. F. Anta, "Bisection (band)width of product networks with application to data centers," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, pp. 570–580, Mar. 2014.

[34] S. Bezrukov, R. Elsaösser, B. Monien, R. Preis, and J. P. Tillich, "New spectral lower bounds on the bisection width of graphs," *Theoretical Comput. Sci.*, vol. 320, pp. 155–174, Jun. 2004.

[35] M. Garey and D. Johnson, "Some simplified NP-complete graph problems," *Theoretical Comput. Sci.*, vol. 1, pp. 237–267, Feb. 1976.

[36] G. Karypis, R. Aggarwal, V. Kumar, and S. Shekhar, "Multilevel hypergraph partitioning: Applications in VLSI domain," in *Proc. Design Automation Conf.*, Jun. 1997, pp. 526–529.

[37] G. Karypis and V. Kumar, "A fast and high quality multilevel scheme for partitioning irregular graphs," *SIAM J. Scientific Computing*, vol. 20, pp. 359–392, Aug. 1998.

[38] K. Efe and A. Fernández, "Products of networks with logarithmic diameter and fixed degree," *IEEE Trans. Parallel Distrib. Syst.*, vol. 6, pp. 963–975, Sep. 1995.

[39] G. D. Vecchia and C. Sanges, "A recursively scalable network VLSI implementation," *Future Generation Computer Systems*, vol. 4, no. 3, pp. 235–243, 1988.

[40] Y. Li, S. Peng, and W. Chu, "Recursive dual-net: A new universal network for supercomputers of the next generation," in *Int. Conf. Algorithms and Archit. for Parallel Process.*, 2009, pp. 809–820.

[41] W. J. Dally, "Performance analysis of k-ary n-cube interconnection networks," *IEEE Trans. Comput.*, vol. 39, pp. 775–785, Jun. 1990.

[42] M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," in *Proc. ACM SIGCOMM Conf.*, Aug. 2008, pp. 63–74.

[43] "June 2017 — TOP500 Supercomputer Sites," https://www.top500.org/lists/2017/06/, 2017.

[44] A. S. Bland, J. C. Wells, O. E. Messer, O. R. Hernandez, and J. H. Rogers, "Titan: Early experience with the Cray XK6 at Oak Ridge National Laboratory," in *Proc. Cray User Group Conf.*, May 2012.

[45] M. Moudi and M. Othman, "The challenge of interconnect topologies to improve communication in supercomputers," in *Proc. Int. Conf. Recent Trends Inform. Process. Computing*, Dec. 2012, pp. 137–144.

[46] H. Casanova, A. Giersch, A. Legrand, M. Quinson, and F. Suter, "Versatile, scalable, and accurate simulation of distributed applications and platforms," *J. Parallel Distrib. Computing*, vol. 74, no. 10, pp. 2899–2917, Jun. 2014. [Online]. Available: http://hal.inria.fr/hal-01017319

[47] L. Bobelin, A. Legrand, M. David, P. Navarro, M. Quinson, F. Sutar, and C. Thiery, "Scalable multi-purpose network representation for large scale distributed system simulation," in *Proc. IEEE/ACM Int. Symp. Cluster, Cloud and Grid Computing*, May 2012, pp. 220–227.

[48] "The NASA Advanced Supercomputing (NAS) Parallel Benchmarks," http://www.nas.nasa.gov/Software/NPB/, 2017.

[49] J. C. Sancho, A. Robles, and J. Duato, "An effective methodology to improve the performance of the up*/down* routing algorithm," *IEEE Trans. Parallel Distrib. Syst.*, vol. 15, no. 8, pp. 740–754, 2004.

[50] M. Ebrahimi and M. Daneshtalab, "EbDa: a new theory on design and verification of deadlock-free interconnection networks," in *Proc. Int. Symp. Comput. Archit.*, Jun. 2017, pp. 703–715.

[51] R. Kawano, R. Yasudo, H. Matsutani, M. Koibuchi, and H. Amano, "HiRy: An advanced theory on design of deadlock-free adaptive routing for arbitrary topologies," in *Proc. Int. Conf. Parallel Distrib. Syst.*, Dec. 2017.

[52] T. Skeie, O. Lysne, J. Flich, P. Lopez, A. Robles, and J. Duato, "LASH-TOR: A generic transition-oriented routing algorithm," in *Proc. Int. Conf. Parallel Distrib. Syst.*, Jul. 2004, pp. 595–604.

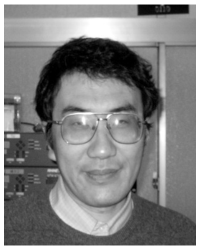
[53] A. Ramrakhyani, P. V. Gratz, and T. Krishna, "Synchronized Progress in Interconnection Networks (SPIN): A new theory for deadlock freedom," Jun. 2018.

Ryota Yasudo received the BE and ME degrees from Keio University, Japan, in 2014 and 2016, respectively. He is currently working towards the PhD degree at Keio University. He was a visiting student at Imperial College London during summer in 2017. His current research interests include topologies and routing methods for interconnection networks, parallel computing, and FPGA-based reconfigurable computing.

Michihiro Koibuchi received the BE, ME, and PhD degrees from Keio University, Japan, in 2000, 2002, and 2003, respectively. He is currently an Associate Professor with the Information Systems Architecture Research Division, National Institute of Informatics, Graduate University for Advanced Studies, Japan. His research interests include the areas of high-performance computing and interconnection networks. He is a member of the IEEE.

Koji Nakano received the BE, ME and PhD degrees from Department of Computer Science, Osaka University, Japan in 1987, 1989, and 1992 respectively. In 1992-1995, he was a Research Scientist at Advanced Research Laboratory. Hitachi Ltd. In 1995, he joined Department of Electrical and Computer Engineering, Nagoya Institute of Technology. In 2001, he moved to School of Information Science, Japan Advanced Institute of Science and Technology, where he was an associate professor. He has been a full professor at School of Engineering, Hiroshima University from 2003. He has published extensively in journals, conference proceedings, and book chapters. He served on the editorial board of journals including IEEE Transactions on Parallel and Distributed Systems, IEICE Transactions on Information and Systems, and International Journal of Foundations on Computer Science. His research interests include image processing, hardware algorithms, GPU-based computing, FPGA-based reconfigurable computing, parallel computing, algorithms and architectures.

Hiroki Matsutani received the BA, ME, and PhD degrees from Keio University, Japan, in 2004, 2006, and 2008, respectively. He is currently an Associate Professor with the Department of Information and Computer Science, Faculty of Science and Technology, Keio University. His research interests include the areas of computer architecture and interconnection networks. He is a member of the IEEE.



Hideharu Amano received the PhD degree from Keio University, Japan, in 1986. He is currently a Professor with Department of Information and Computer Science, Faculty of Science and Technology, Keio University. His research interests include the areas of parallel processing and reconfigurable systems. He is a member of the IEEE.