Theoretical Design Methodology for Practical Interconnection Networks

Ryota Yasudo

A dissertation submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

School of Science for Open and Environmental Systems
Graduate School of Science and Technology
Keio University

January 2019

Abstract

End-to-end network latency is a concern for parallel applications in high-performance computing platforms and high-end data centers. On one hand, computer architects have tried to design interconnection networks experimentally and empirically. On the other hand, theorists have tried to model computer networks and then studied their properties theoretically. However, the model does not sufficiently capture real computer systems. This dissertation aims at establishing a novel method for designing high-performance network topologies to bridge a gap between the theoretical and practical studies. In particular, we make use of the knowledge of graph theory, network science, and design theory, as well as the research on interconnection networks.

We firstly 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 then 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, Dragonfly, and Fat-tree, our networks attain higher performance and smaller power and costs.

Furthermore, we propose adding ports to a host as a method for reducing the network latency as well as increasing the number of ports of a switch. To this end, we extend a host-switch graph so that it represents multi-port hosts. Multi-port hosts are conventionally used for link aggregation (LA) and network duplication (ND), but they do not reduce the hop count. We hence propose the permutation of host-switch mapping for reducing the hop count. It can be applied to LA and ND, and we label the obtained networks p-LA and p-ND, respectively. In addition, we propose the application of a finite projective plane (an instance of block designs) and label it PP. Our methods can be applied to arbitrary topologies, and thus we can directly use any existing topologies. We evaluate five designs above (LA, ND, p-LA, p-ND, and PP) for randomly-optimized, torus, Dragonfly, and Fat-tree topologies in terms of the design complexity, the hop count, the bisection width, costs, the size of routing tables, and simulated message passing interface (MPI) performance. Our results demonstrate that our methods (p-LA, p-ND, and PP) reduce the hop count while increasing the bisection width and costs for every topology. In particular, we demonstrate that PP is a cost-effective method for reducing the hop count, especially for randomly optimized and Fat-tree topologies.

Acknowledgments

First and foremost, I would like to thank my academic adviser, Professor Hideharu Amano, for his tremendous commitment to teaching and mentoring throughout my career as a graduate student.

I would like to thank Associate Professor Michihiro Koibuchi for his teaching and mentoring since I became a research assistant at National Institute of Informatics.

I would also like to thank Professor Tadao Nakamura, Professor Koji Nakano, and Associate Professor Hiroki Matsutani. I was fortunate to have them as teachers and collaborators, and my scientific perspective broadened.

I would also like to thank Professor Wayne Luk and Dr. Jose Gabriel de Figueiredo Coutinho for their tremendous supports when I was visiting Imperial College London, as well as collaborative research on reconfigurable computing.

I am grateful to my doctoral committee members, Professor Koji Nakano, Professor Katsuhiro Ota, and Associate Professor Hiroki Matsutani for their careful reviews and valuable comments to my thesis.

Last but not least, I would like to thank my family and friends for their supports and encouragement during my time at Keio University.

Ryota Yasudo
Yokohama, Japan
December 2018

Contents

List of Tables

List of Figures

List of Theorems

Chapter 1

Introduction

1.1 Motivation

The international roadmap for devices and systems (IRDS 2017 edition [7]) predicts that a cloud system is one of the important market drivers. Cloud systems support many important applications such as web service, multimedia, shopping, big data analytics, and high-performance scientific computation. In particular, big data analytics is increasingly important with the growth of big data for social networking, artificial intelligence (AI), smart cities, and so forth. Big data requires abundant computing power and continuing performance scaling.

A typical application of big data analytics is used as the Graph 500 benchmark [4]. In the Graph 500 benchmark, graph construction and breadth-first search (BFS) are processed. BFS requires many communications as compared with computation. Furthermore, the communications have few locality. To accelerate the Graph 500 benchmark, reducing costs of data movement is therefore essential.

As with big data applications including the Graph 500 benchmark, data-intensive applications such as physical system simulation are also emerging. In general, Peter Kogge suggests that we are now facing the *locality wall* on the heels of the memory wall and the power wall; growing non-predictable regularity and non-locality limits the performance [2]. To accelerate such applications, we need to improve the latency and the bandwidth for both memory and interconnection networks. Especially, interconnection networks should provide faster all-to-all communication and support irregular communication patterns. Thus, irregular network topology that handles non-local traffic would work effectively.

1.2 Objectives

A long-standing design goal for high-performance computing (HPC) is to provide low end-to-end network latencies between compute nodes. This requirement for low latency is also relevant for high-end data center networks (DCN). For example, DCNs that target high-frequency trading (HFT) applications can benefit from end-to-end latencies down to microseconds levels, thus motivating the use of interconnects traditionally used in HPC platforms.

Supporting large-scale applications requires large-scale platforms, e.g., exascale platforms that aggregate millions of cores in hundreds of thousands of compute nodes. Large-scale HPC platforms are currently deployed as compute nodes that are interconnected using large numbers of switches, and DCNs are built following a hierarchical structure with so-called top of rack (ToR) switches, cluster routers, and border routers [9]. In both cases, end-to-end network paths between two compute nodes traverse multiple switches located in different cabinets. End-to-end latencies must decrease to design scalable platforms for workloads that lead to many small message exchanges between compute nodes. Especially, switch delays are high as compared with wire and flit injection delays; for instance, port-to-port switch latency reaches 90 nanoseconds in InfiniBand EDR 100 Gb/s switch [6]. Thus, the number of switches traversed by a network path, called the *hop count*, should be reduced.

Recent research shows that complex networks such as small-world and random networks have low hop counts. Moreover, such complex networks should provide fast all-to-all communications and support irregular traffic patterns, and thus they satisfy the requirement of data-intensive applications, described in Section 1.1. Therefore, this dissertations study a design method and practical feasibility of complex network topologies with low hop count.

1.3 Contributions

To achieve the objectives above, this dissertation mainly makes the following contributions.

1. We survey both theoretical and practical studies for graphs, complex networks, and interconnection networks, including graph theory, network science, design theory, and computer engineering.

2. We propose a novel graph called a *host-switch graph*. It will firstly be defined as a model of an interconnection network with single-port hosts, and then extended to a model of an interconnection network with multi-port hosts.

3. We establish two novel graph problems: the *radix/diameter problem* (*RDP*) and the *order/radix problem* (*ORP*). Subsequently, we provide some theory and solutions for both problems.

4. We propose and run a heuristic algorithm for solving ORP. It is based on the simulated annealing, but it includes a new concept called a *2-swing* operation and a *continuous Moore bound*.

5. For interconnection networks with multi-port hosts, we propose two novel methods for reducing the network latency: *permutation of host-switch mapping* and application of finite projective planes.

6. We compare our proposed network topologies with existing network topologies including theoretically proposed topologies and practical topologies used in supercomputers ranked in TOP500.

1.4 Dissertation Outline

The rest of this dissertation is organized as follows.

Chapter 2 describes technical and theoretical backgrounds. Technical backgrounds include architectural concepts of interconnection networks for large-scale parallel computer systems from three perspectives: the topology, the routing, and the layout. Theoretical backgrounds include graph theory, design theory, and network science; the study set forth in this dissertation effectively applies them to practical interconnection networks.

Chapter 3 studies low-latency interconnection networks with single-port hosts. The two novel concepts are proposed: a host-switch graph and the order/degree problem. A host-switch graph is a model of an interconnection networks with hosts and switches, and the order/degree problem is a graph problem for minimizing the ideal (zero-load) end-to-end latency of an interconnection network. Several interesting findings are described.

Chapter 4 studies low-latency interconnection networks with multi-port hosts. To represent a network with multi-port hosts, a host-switch graph is extended. As in Chapter 3, the ideal end-to-end latency is minimized. To this end, the solution obtained in Chapter 3 can be utilized. It is shown that design theory provides excellent design of the networks with multi-port hosts.

Chapter 5 summarizes this dissertation and suggests future directions.

Chapter 2

Technical and Theoretical Backgrounds

2.1 Interconnection Networks

An *interconnection network* is a programmable system that transports data between terminals [39]. It occurs at many scales, including on-chip networks (a.k.a. networks-on-chip) and off-chip networks. Physical characteristics depend on the scale, but fundamental principles are the same. In this section, we describe interconnection networks with a focus on off-chip ones.

Three issues mainly dominate the design of an interconnection network: topology, switching technique, and routing algorithm. In addition to the three issues above, this section studies the physical layout, which is becoming more and more important due to the increasing gap between switching delays and cable delays.

2.1.1 Topology

2.1.1.1 Theoretical Studies on Network Topology

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 analyzing 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 [86]—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 [21, 77, 87]. Figure 2.1 illustrates a concept of theoretical studies of network topology.

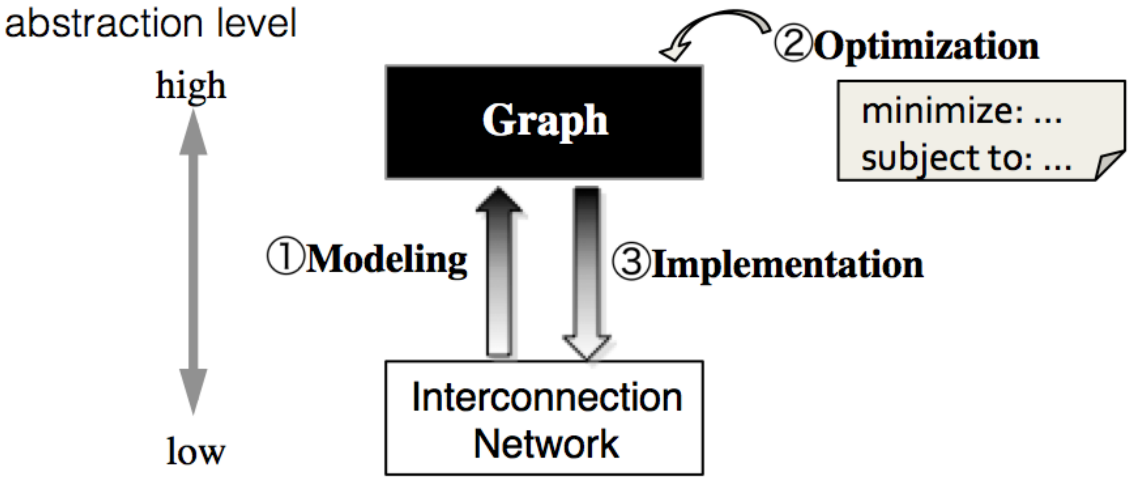


Figure 2.1: Concept of theoretical studies of network topology.

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 [28], the Erdős-Rényi model (random graph) [49], and the Watts-Strogatz model (small-world networks) [121]. Some solutions for the ODP are such complex graphs and applied to computer systems, including high-performance computing systems [76], data centers [110], and on-chip networks [95]. To apply such complex topologies to practical networks, physical layouts [74] and routing algorithms [55] 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 3.4). Therefore, we should radically change both a model of interconnection networks and a graph problem.

2.1.1.2 Practical Studies on Network Topology

Practical studies on topologies of interconnection networks for parallel/distributed computer systems have a long history. In the 1970s, hypercubes were used in many systems such as Cosmic Cube [106];

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 [71] are researched for reducing communication overhead; and now, in the 2010s, the high-radix networks are used in commercial high-performance computers [13, 15].

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 [78] is widely used in parallel/distributed computer systems from generation to generation, though technology for each generation is different (e.g., both CM-5 [63] in the 1980s and Tianhe-2 [81] 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.

2.1.1.3 Commonly Used Existing Topologies

Torus The torus topology is a mesh graph (a.k.a. a lattice graph or a grid graph) with wrap-around channels, and consequently each dimension constitutes a ring structure. Formally, a K-ary N-torus is a topology with parameters K and N. Each node is identified by a N-bit base-K address $a_{N-1}a_{N-2}\cdots a_0$, and connected to nodes 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 $j \neq i$).

Dragonfly Dragonfly is, so to speak, a meta-topology proposed for designing technology-driven, highly-scalable interconnection networks with high-radix routers [71]. It consists of multiple *group*s, in which an intra-group network connect the routers, and an inter-group network as shown in Figure 2.2. In [71], Kim *et al.* suggest that every intra-group network should be a clique or Flattened butterfly and the inter-group network should be a clique. However, rather interestingly, the topologies of inter- and intra-group networks are not deterministic since Dragonfly is not topology itself but a meta-topology. Also, we can change parameters such as the number of links within a group and between groups.

Dragonfly provides a high-performance networks, so it is adopted by many supercomputers ranked in Top 500.

Fat-tree Fat-tree is a class of *bidirectional multi-stage interconnection networks* (*BMIN*). MIN is a directed network consisting of multiple stages as shown in Figure 2.3; then, by folding it, we get an

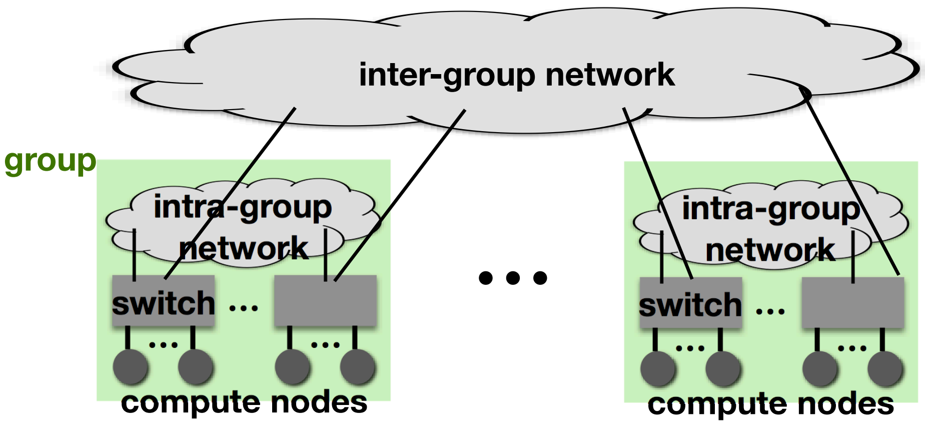


Figure 2.2: A conceptual figure of Dragonfly.

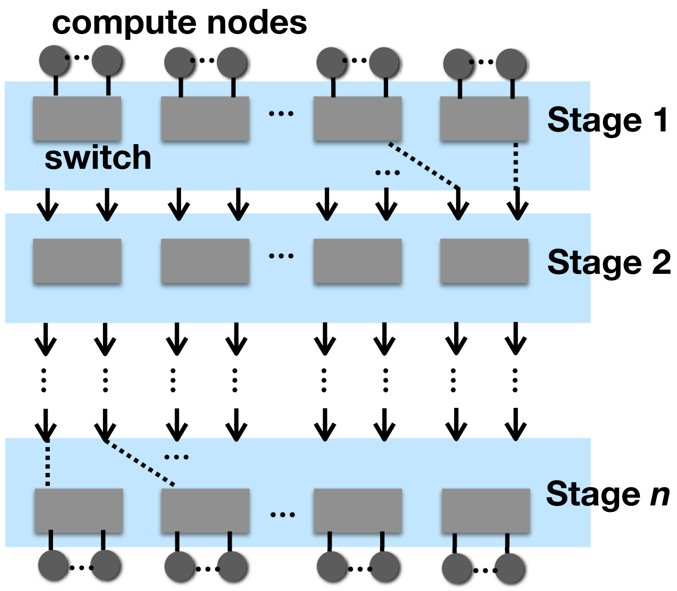


Figure 2.3: A conceptual figure of a multi-stage interconnection network.

undirected network, i.e., BMIN. An example of Fat-tree is shown in Figure 2.4. In this figure, Fat-tree has three layers—called core, aggregation, and edge layers—and it provides full bisection bandwidth.

2.1.1.4 Bridging a Gap between Theoretical and Practical Studies

The study set forth in this dissertation 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. There exist many gaps between graphs and real systems; a real system may consist of various types of nodes, including CPUs, accelerators, memories, and switches, and we should consider the execution time for real applications, the fault tolerance, the layout, and the energy consumption.

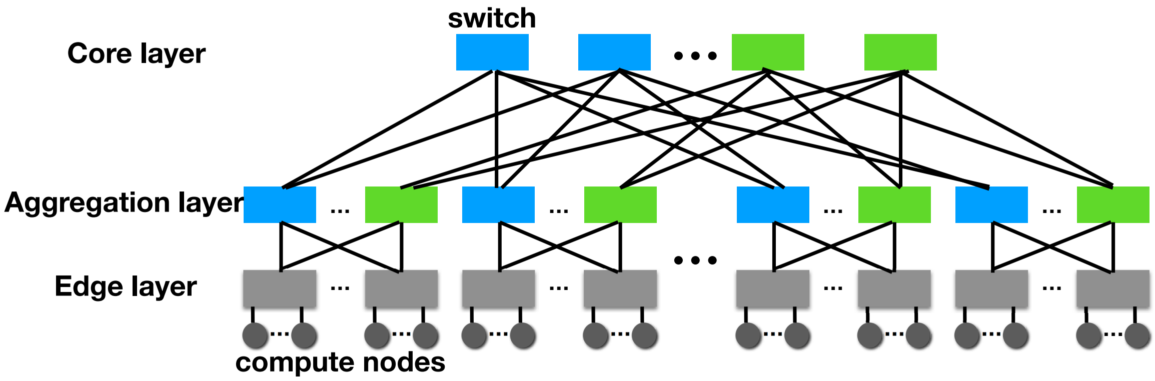


Figure 2.4: An example of Fat-tree.

This study focuses on distinguishing hosts and switches and considering the end-to-end latency between two hosts. We will present a novel graph called a *host-switch graph*, which consists of *host* vertices and *switch* vertices. 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.

2.1.2 Routing

2.1.2.1 Switch

Large-scale HPC platforms are currently deployed as compute nodes that are interconnected using large numbers of switches, and DCNs are built following a hierarchical structure with so-called top of rack (ToR) switches, cluster routers, and border routers [9]. In both cases, end-to-end network paths between two compute nodes traverse multiple switches located in different cabinets. End-to-end latencies must decrease to design scalable platforms for workloads that lead to many small message exchanges between compute nodes. Especially, switch delays are high as compared with wire and flit injection delays; for instance, port-to-port switch latency reaches 100 nanoseconds in InfiniBand QDR [6]. Thus, the number of switches traversed by a network path, called the hop count, should be reduced.

To reduce the hop count, switches with dozens of ports, called high-radix switches, have been used in the HPC domain for many years [72, 104]. High-radix switches expand the design space for topologies because a variety of high-degree topologies become feasible. It is thus possible to use network topologies with the low diameter and the low average shortest path length (ASPL) [21, 56, 67, 71, 77, 87], both measured in the hop count. Recently, random [76, 110] (or randomly optimized [90, 124]) topologies are demonstrated to provide the ASPL and the diameter that are close to the lower bounds [90, 124]. However, switch complexity increases quadratically impacting on the area, costs, and the latency in a switch. Thus, the number of ports of a switch is limited, and low-radix switches are preferred if the hop count is the same [123].

2.1.2.2 Routing Algorithm

A switch determines a route (a proper output port) for each incoming packet so that it finally reaches the destination node. In general, this *routing* must guarantee *deadlock-freedom* and *livelock-freedom*. A *deadlock* refers to a situation such that group of agents (packets) are unable to act because of waiting each other to release some resource (buffers or channels). A *livelock* refers to a situation such that packets are unable to reach their destinations although they can continue to move. Obviously, a livelock occurs only if non-minimal routing is used. A livelock can easily be avoided by specifying one or more paths for every pair of nodes and bounding the number of misroutings. However, how to design deadlock-free routing is not clear.

For regular topologies, specific deadlock-free routing algorithms can be designed and used; for example, dimension-order routing for two-dimensional mesh topologies. However, for irregular topologies, there exist no specific routing algorithm, and hence we need a general method to design deadlock-free routing algorithm for arbitrary topologies. We now describe previous work that proposes such methods and theoretical foundations on deadlock-free routing.

Dally's theory Dally *et al.* introduces a *channel dependency graph* (*CDG*) for guaranteeing deadlock-freedom of arbitrary interconnection networks [38]. A *CDG*, for a given interconnection network and a given set of possible routing paths, is a directed graph where

- the vertices denote the channels of the interconnection network;
- the edges denote the possible routing paths.

Dally *et al.* then proved that an interconnection network is deadlock-free if and only if the CDG is acyclic (we call it *Dally's theory*). Note that Dally's theory per se is not routing algorithm. Also, Dally's theory allows us only to check whether given pair of a network and routing is deadlock-free or not.

Turn model and its extensions The *turn model* [59] is a well-known application of Dally's theory for two-dimensional mesh/torus topologies. In the turn model, the packet forwardings are divided into four directions: north, south, east, and west. The change of directions, called a *turn*, are then classified into eight patterns; note that all the turns make right angles. As a result, we can see there exist only two types of cyclic dependencies (clockwise and counterclockwise). Thus, a deadlock-free routing can be designed by prohibiting one turn for each type of cyclic dependency.

The turn model, however, restricts the topology to mesh and tori. Thus, the turn model has been extended to n-dimensional topologies without diagonal links [45] and arbitrary topologies [69].

Using additional buffers Virtual channels are also useful for avoiding deadlock on the basis of Dally's theory. The *layered shortest path* (*LASH*) [113] uses virtual channels to achieve deadlock-free minimal routing. LASH can be combined with the *transition oriented routing* (*TOR*) [102] to enable flexibility of the different algorithms among the virtual channel layers; it is called *LASH-TOR* [112]. *In-transit buffers* (*ITB*) also enable a deadlock-free minimal routing. Packets are ejected from the network temporarily and stored in an ITB when deadlock occurs. These methods can be applied to arbitrary network topology and use the shortest path routing. Thus, we can assume the shortest path routing for any topology.

Up*/Down* routing *Up*/Down* routing*, introduced in [103], is a deadlock-free routing method that avoids cyclic dependencies by using spanning trees obtained by using a breadth-first search. Given network topology, a spanning tree is constructed and the packet forwardings are classified into the *up* direction or the *down* direction; then, a routing is deadlock-free if there exist no move to the up direction after moves to the down direction. A drawback of the Up*/Down* routing is the poor performance due to the biased link utilization.

Several methods for improving Up*/Down* routing have been proposed. Sancho *et al.* demonstrated that using a depth-first search (DFS) instead of BFS when constructing a spanning tree improves traffic balance [100]. *Left-up-first turn routing* (*L-turn routing*) [75] improves Up*/Down* routing by using *L-R directed-graph*, which is a transformation of the spanning tree and has left and right directions in addition to up and down directions. Using this graph, it distributes prohibited turns (in particular, avoids the heavy traffic around the root node of the spanning tree), and consequently the throughput is improved.

2.1.3 Layout

The nodes of an interconnection network are typically arranged in two-dimensional space. The layout determines the lengths of cables, and consequently it is essential for reducing signal propagation delay and cabling costs. Large-scale computer systems such as supercomputers have historically used Manhattan cabling in floorplan, because Euclid cabling makes both cabling and its maintenance become complex due to the diagonal cables on a floor. In the case of simple networks such as 2-D mesh networks, the layout is easily determined. However, in the case of irregular networks studied in this dissertation, the layout is not clear. Thus, we focus on the layout of irregular network topologies.

2.1.3.1 Quadratic Assignment Problem

The layout of an interconnection network is determined by mapping each cluster to a cabinet on a floorplan so that the total cable length becomes minimum. This problem corresponds to the facility location problem, which has been studied in operation research [52]. This problem is NP-hard, and thus there is no known algorithm for solving this problem in polynomial time. Hence, metaheuristicsbased techniques have been developed to solve it. Solutions to this problem have been used in the computer industry for computer chip design and physical network layouts for HPC systems.

When we determine physical network layouts, we should reduce the cable lengths (the maximum length or the average length or both); this should be formulated as a quadratic assignment problem (QAP). It is reported that several algorithms can successfully applied to QAP (e.g., the simulated annealing [35], the robust tabu search [117], the reactive tabu search [20], the greedy randomized adaptive search procedure [96], the fast ant colony algorithm (FANT) [116], and the memetic algorithm [84].

2.1.3.2 Randomly Optimized Grid Graph

Nakano *et al.* propose a method for designing interconnection networks with link-length constraints [90] by introducing a new graph called a *grid graph*. Using this, they propose a randomized algorithm for optimizing network topologies and layouts at the same time. A *grid graph* is a graph $G = (V, E)$ such that $V = \{(x, y) \mid 0 \leqslant x, y \leqslant \sqrt{N} - 1\}$ is a set of N nodes and E is a set of edges connecting a pair of two distinct nodes in V. We can think that nodes in V are arranged in a 2-dimensional space so that each node (x, y) is located at position (x, y). Let $l(u, v)$ denote the Manhattan distance of two nodes u and v in V, that is, $l(u, v) = |u_x - v_x| + |u_y - v_y|$, where $u = (u_x, u_y)$ and $v = (v_x, v_y)$. In a network with topology represented by a grid graph, the two nodes (u, v) are connected by a communication link of length $l(u, v)$ wired along the grid.

A grid graph $G = (V, E)$ is *L-restricted* if $l(u, v) \leqslant L$ for all edges $(u, v) \in E$. Clearly, in a network with topology represented by an L-restricted grid graph, the length of every communication link is restricted to no more than L. A grid graph is *K-regular* if every node is connected with K edges. In [90], the authors show lower bounds on the diameter and the average shortest path length (ASPL) of K-regular L-restricted grid graph and provide a K-regular L-restricted grid graph whose diameter and ASPL are close to the lower bounds by using a randomized algorithm.

Moreover, [90] discusses quite interesting relationship between N, K, and L; the authors derive the following asymptotic formula, which provides a *well-balanced*[1] grid graph:

$$\Theta(\log N / \log K) \quad \approx \quad \Theta(\sqrt{N}/L). \tag{2.1}$$

Let us here describe an interesting finding. From (2.1), we have a decreasing function $\log K = \Theta(L \log N/\sqrt{N})$ of N. Thus, if L is fixed and N is increased, K must be decreased to keep well-balanced. Quite surprisingly, this relationship suggests that we should reduce the number of ports in each node of a computer system when we increase the number of nodes, provided that we use communication cables with the same technology.

[1] A K-regular L-restricted grid graph is *well-balanced* if the absolute difference between the lower bound on the ASPL of K-regular grid graph and that of L-restricted grid graph is a local minimum.

Nakahara *et al.* extended a grid graph so that it represents a 3D-NoC and call it a *stacked grid graph* [89]. They proposed a method for optimizing the average shortest path length and energy consumption of a 3D-NoC by using a multi-objective simulated annealing (MOSA) [92].

2.2 An Undirected Graph

2.2.1 Definition and Notation

An *undirected graph* is an ordered pair $G = (V, E)$ where

- V is a set of elements called *vertices*, and
- E is a set of elements called edges, each of which is a 2-element subset of V.

In the field of computer architecture, an undirected graph is used to represent a network of a computer system. An undirected graph is preferred to a directed graph because an interconnection networks use bi-directional links rather than uni-directinal ones.

There are three parameters important for interconnection networks: the number n of vertices (called the *order*), the maximum number of edges connected to a vertex Δ (called the *degree*), and the maximum value of the shortest path length D (called the *diameter*). The shortest path length $\ell(u, v)$ is the smallest possible path length between two vertices, u and v. The order should increase so that many processing units operate in parallel to improve the performance. The degree should be limited because designing a switch with many ports require high costs and the switching latency. The diameter should be reduced as much as possible because the ideal[2] communication latency depends on the path length between a source and a destination; the diameter is especially important for reducing the worst case communication latency.

In this context, the *degree/diameter problem* (*DDP*) has traditionally been attracting attention. This problem is defined as follows.

Problem 2.1 (Degree/Diameter Problem)**.** *Given natural numbers Δ and D, find the largest possible order n in an undirected graph with maximum degree Δ and diameter D.*

The DDP is considered for several classes of graphs, including undirected graphs, directed graphs, mixed graphs, bipartite graphs, planar graphs, and so forth. Among them, this dissertation focuses on the cases of undirected graphs, which are used for representing interconnection networks, and bipartite undirected graphs, which we will use in Chapter 4.

2.2.2 Degree/Diameter Problem for General Graphs

Let us consider the tight upper bound on the order n^+ of an undirected graph $G = (V, E)$ with degree Δ and diameter D. Trivially, if $\Delta = 1$, then $D = 1$ and $n^+ = 2$. Hence, we assume that $\Delta \geqslant 2$.

[2] The ideal communication latency means the communication latency if there is no packet congestion.

For any fixed vertex $u \in V$, we can partition all the vertices in V into subsets $V_0, V_1, \ldots$ such that $V_i = \{v \in V \mid \ell(u,v) = i\}$. Clearly, $V_0 = \{u\}$ and $|V_0| = 1$ hold. Since u is connected with at most Δ edges, $|V_1| \leqslant \Delta$ holds. Since each vertex in V_1 is connected with at most $\Delta - 1$ vertices in V_2, $|V_2| \leqslant \Delta(\Delta - 1)$ holds. In general, we have

$$|V_i| \leqslant \begin{cases} 1 & \text{if } i = 0 \\ \Delta(\Delta-1)^{i-1} & \text{if } i \geqslant 1 \end{cases}. \tag{2.2}$$

Thus, the upper bound on the order of an undirected graph with degree Δ and diameter D becomes as follows:

$$n^+ = 1 + \sum_{i=1}^{D} \Delta(\Delta-1)^{i-1} \tag{2.3}$$

$$= \begin{cases} 1 + \Delta\frac{(\Delta-1)^D - 1}{\Delta-2} & \text{if } \Delta > 2 \\ 2D + 1 & \text{if } \Delta = 2 \end{cases}. \tag{2.4}$$

This upper bound is called the *Moore bound*, and a graph of order n^+ is called a *Moore graph* [64].

Solutions of DDP are applied to topologies of interconnection networks. For example, MMS graphs [83] are applied to Slim Fly [21].

2.2.3 Degree/Diameter Problem for Bipartite Graphs

A *bipartite graph* (a.k.a. *bigraph*) is a graph $G = (V_1, V_2, E)$ where

- V_1 and V_2 are two disjoint and independent sets of vertices, and
- E is a set of edges that connect a vertex in V_1 to one in V_2.

A bipartite graph is said to be *biregular* if two vertices in the same bipartition class have the same degree. We can consider DDP limited to the bipartite graphs.

The tight upper bound on the order n^+_{bi} of a bipartite graph with maximum degree Δ and diameter D was given by Biggs [24]:

$$n^+_{\mathrm{bi}} = \begin{cases} \frac{2(\Delta-1)^D - 1}{\Delta-2} & \text{if } \Delta > 2 \\ 2D & \text{if } \Delta = 2 \end{cases}. \tag{2.5}$$

This upper bound is called the *bipartite Moore bound*, and a bipartite graph of order n^+_{bi} is called a *bipartite Moore graph*.

A *generalized polygon* is a biregular bipartite graph such that the girth is equal to $2D$. Feit and Higman proved that a Δ-regular finite generalized polygon (i.e., a bipartite Moore graph) with $\Delta > 2$ is either a complete bipartite graph ($D = 2$), a finite projective plane ($D = 3$), a finite generalized quadrangle ($D = 4$), or a finite generalized hexagon ($D = 6$) [53]. Chapter 4 will apply this theorem for designing interconnection networks.

2.2.4 Order/Degree Problem

Even though DDP solutions have been applied to topologies of interconnection networks, they may not directly be usable for network topologies in supercomputer and data center systems. This is because they are for particular number of vertices (corresponding to compute nodes in a system), whereas the number of nodes in a real system is determined based on practical considerations such as power consumption and costs.

In this context, researchers on computer engineering proposed another graph problem called the *order/degree problem* (*ORP*) [5].

Problem 2.2 (Order/Degree Problem)**.** *Given natural number n and Δ, find the minimum possible diameter D in an undirected graph with order n and maximum degree Δ. If two or more graphs take the minimum diameter, find the minimum possible average shortest path length (ASPL) in an undirected graph with the minimum diameter.*

Note that, by definition, ORP contains two objective functions: the diameter and the ASPL.

Let us consider the tight lower bounds on the diameter D^- and the ASPL A^- of an undirected graph $G = (V, E)$ with order n and maximum degree Δ. For any fixed vertex $u \in V$, we can partition all the vertices in V into $V_0, V_1, \dots$ such that $V_i = \{v \in V \mid \ell(u, v) = i\}$. Clearly, $V_0 = \{u\}$ and $|V_0| = 1$ hold. Since u is connected with at most Δ edges, $|V_1| \leqslant \Delta$ holds. Since each vertex in $|V_1|$ is connected with at most $\Delta - 1$ vertices in V_2, $|V_2| \leqslant \Delta(\Delta - 1)$ holds. In general, we have

$$|V_i| \leqslant \begin{cases} 1 & \text{if } i = 0 \\ \Delta(\Delta-1)^{i-1} & \text{if } i \geqslant 1 \end{cases}. \tag{2.6}$$

From Eq. 2.4, we have

$$D = \begin{cases} \log_{\Delta-1}\left(\frac{(n^+-1)(\Delta-2)}{\Delta} + 1\right) & \text{if } \Delta > 2 \\ \frac{n^+-1}{2} & \text{if } \Delta = 2 \end{cases}. \tag{2.7}$$

Thus, we have

$$D^- = \begin{cases} \left\lceil \log_{\Delta-1}\left(\frac{(n-1)(\Delta-2)}{\Delta} + 1\right) \right\rceil & \text{if } \Delta > 2 \\ \left\lceil \frac{n-1}{2} \right\rceil & \text{if } \Delta = 2 \end{cases}. \tag{2.8}$$

Let $m(i)$ be the *Moore function* such that

$$m(i) = \begin{cases} 1 & \text{if } i = 0 \\ \min\left(1 + \sum_{j=1}^{i} \Delta(\Delta-1)^{j-1}, n\right) & \text{if } i \geqslant 1 \end{cases}. \tag{2.9}$$

Clearly, the number of vertices reachable in i hops from u does not exceed $m(i)$. Thus, we have

$$A^- = \sum_{i \geqslant 1} \frac{(m(i) - m(i-1)) \cdot i}{n-1}. \tag{2.10}$$

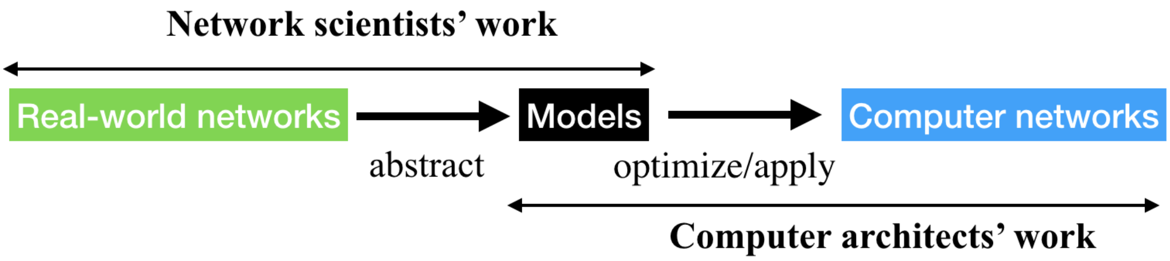


Figure 2.5: Concept of the use of network science for designing computer architecture. Network scientists mathematically model real-world networks, and then computer architects can apply the models for designing computer networks.

2.3 Network Science and its Applications

Thus far, we have described (classical) graph theory and examples of mathematical problems and results. What we have shown in this dissertation is just a tip of an iceberg, and there exists much beautiful and elegant work in the field of graph theory. However, several researchers in various fields such as complex systems, sociology, biology, and computer engineering should understand properties of real-world networks, which are possibly complex, dynamic, and/or stochastic. Thus, the new science of networks called *network science* has been studied empirically as well as theoretically. Mainly, researchers have proposed three basic models of real-world networks, which we describe below. Computer architects utilize network science for designing computer architecture on the basis of the concept shown in Fig. 2.5.

2.3.1 Random Graph Models

A *random graph*—introduced by Solomonoff and Rapoport [114] and studied extensively by Erdős and Rényi [49, 50]—is a graph obtained by randomly sampling from a collection of possible graphs with fixed number of vertices. Among several proposed models, the *Erdős-Rényi model*, described below, is mostly be studied.

The Erdős-Rényi (ER) model generates a random graph $G_{n,p}$ as follows:

1. Fix the number n of vertices and probability p $(0 < p < 1)$.

2. Connect each pair of vertices with independent probability p.

As a result, the ER model generates either of possible $2^{n(n-1)/2}$ graphs, including a complete graph and a graph with no edge. However, it is known that p determines the properties of ER model. For example, the ER model almost surely provides a connected graph when $p \geqslant \log n/n$ while it does not when $p < \log n/n$. Such a qualitative change according to the value of p is called a *phase transition*.

According to [27], the average distance L satisfies

$$L \approx \frac{\log n}{\log\langle k\rangle}, \tag{2.11}$$

where $\langle k \rangle$ denotes the average degree. Thus, when we use a random graph for the network topology, the ASPL becomes $\mathscr{O}(\log n)$, which grows slowly as the order increases. That is why using randomness when constructing network topologies is helpful for designing low-latency interconnection networks.

The random graph $G_{n,p}$ has a specific degree distribution. Since each vertex is connected to the other $n-1$ vertices with probability p, the degree distribution becomes

$$p(k) = \frac{(n-1)!}{k!(n-1-k)!} p^k (1-p)^{n-1-k}, \tag{2.12}$$

where $p(k)$ denotes the probability that the degree of a vertex is k. It is clearly a binomial distribution and becomes the Poisson distribution in the limit where $n \to \infty$, $p \to 0$, and $(n-1)p \to \lambda$ (λ is a positive constant).

Computer architects may think that the degree distribution should not be the Poisson distribution. Instead, a network should be regular, i.e., the degree is fixed. To construct a random network with arbitrary degree distribution, we can use an extended model of random graphs. The *configuration model* [93] generates a random graph with arbitrary degree distribution as follows:

1. Fix the number n of vertices.
2. Fix the degree distribution $p(0), p(1), \ldots, p(k_{\max})$ where $k_{\max}$ denotes the maximum degree ($k_{\max} \leqslant n-1$).
3. Generate the *degree sequence* $k_1, k_2, \ldots k_n$ so that it satisfies the fixed degree distribution.
4. Generate a random graph $G = (V, E)$ where $V = \{v_1, v_2, \ldots v_n\}$ and the degree of v_i is k_i.

According to [93], the average distance L of a configuration model satisfies the following if $\langle k^2 \rangle$ exists in the limit where $n \to \infty$:

$$L = 1 + \frac{\log \frac{n}{\langle k \rangle}}{\log \frac{\langle k^2 \rangle - \langle k \rangle}{\langle k \rangle}}. \tag{2.13}$$

One of the important examples of the configuration model is a *regular random graph*. It is a random graph where the degree of all the vertices is the same, i.e., $\langle k \rangle$. Thus, it is straightforward to use a regular random graph for a topology of an interconnection network since the network typically consists of the switches with fixed ports. Several researchers propose the use of a regular random graph for designing network topologies [51, 74, 90, 109, 110, 123, 124].

2.3.2 Watts-Strogatz Model

In 1967, Milgram demonstrated the *small-world phenomenon*, that is to say, human society consists of a network with short path-lengths; more specifically, people in the United States seemed to be connected by approximately three friendship links on average, without speculating on global linkages [85]. This phenomenon is sometimes associated with the phrase "six degrees of separation" and regarded as a characteristics of real-world complex networks.

Afterward, in 1998, Wattz and Strogatz published an epoch-making paper proposing the *Wattz-Strogatz (WS) model* (a.k.a. the small-world model), which excellently characterizes the networks with the small-world phenomenon [121]. The WS model provides a network with the short average distance and the large *clustering coefficient*, which denotes the probability that two adjacent nodes of a node is connected.

The WS model provides a network as follows:

1. Fix the number n of vertices and the average degree $\langle k \rangle$ ($\langle k \rangle$ is an even number).
2. Construct a cycle graph with n vertices.
3. Connect a vertex u and the vertices that can be reached within $\langle k \rangle/2$ hops from u.
4. Change an endpoint of an edge with independent probability p ($0 \leqslant p \leqslant 1$).

Obviously, the obtained network depends on the value of p. When $p = 0$, a network becomes an extended cycle graph. When $p = 1$, a network almost corresponds to a random graph obtained by the ER model. The small-world phenomenon is obtained when $p \in [0.01, 0.1]$. The degree of the WS model is fixed when $p = 0$, and its distribution approaches the Poisson distribution as p increases.

According to [19], the clustering coefficient $C(p)$ of the WS model with probability p satisfies

$$C(p) = \frac{3\langle k \rangle - 6}{4\langle k \rangle - 4}(1-p)^3. \tag{2.14}$$

According to [94], the average distance $L(p)$ of the WS model with probability p satisfies

$$\begin{aligned} L(p) &= \frac{2n}{\langle k \rangle} f\left(\frac{n\langle k \rangle p}{2}\right), & (2.15) \\ f(x) &= \frac{1}{2\sqrt{x^2+2x}} \tanh^{-1}\left(\frac{x}{\sqrt{x^2+2x}}\right). & (2.16) \end{aligned}$$

Note that $L(p)$ is minimized when $p = 1$. Hence we should use the ER model rather than the WS model if we consider only the ASPL of a network topology. However, the clustering coefficient decreases as p increases.

Several researchers propose the use of a small-world network for designing network topologies [41, 95, 108]

2.3.3 Models of Scale-free Networks

One might wonder if the models described above capture real-world networks precisely. In fact, they do not. In 1999, Albert et al. showed that the power law described the topology of the World-Wide Web [14]. Formally, the degree distribution $p(k)$ satisfies

$$p(k) \propto k^{-\gamma}. \tag{2.17}$$

A random graph with the power law degree distribution is called a *scale-free* network. There are many models describing the scale-free networks. For example, the configuration model, described earlier, is one of such models.

Cohen and Havlin [34] shew that the average distance L (or the diameter) of the scale-free networks satisfies

$$L \propto \begin{cases} \log \log n & \text{if } 2 < \gamma < 3 \\ \log n / \log \log n & \text{if } \gamma = 3. \end{cases} \tag{2.18}$$

Note that the average distance grows more slowly as n increases than it does in the case of small-world networks. Thus, Cohen and Havlin say that scale-free networks are *ultra-small*.

Several researchers propose the use of a scale-free network for designing network topologies [54, 62]. However, a scale-free network would not be useful for designing interconnection networks because it requires switches with many ports, which should induce high switching latency and costs. Also, the number of ports of a switch do not vary.

2.4 Design Theory

In combinatorial mathematics, design theory [43] refers to the study of *designs*—systems of finite sets whose arrangements satisfy generalized concepts of balance or symmetry or both. In particular, it studies necessary and sufficient conditions for the existence of a *block design*.

A *block design* with parameters (v, b, r, k, λ) is a pair $(\mathbf{X}, \mathcal{A})$ where

- $\mathbf{X}$ is a set of v elements (called *points*),
- $\mathcal{A}$ is a family of b subsets of $\mathbf{X}$, each of cardinality k (called *blocks*),
- Every point occurs in exactly r blocks, and
- Every pair of distinct points occurs in exactly λ blocks.

If $\mathcal{A} = \{\mathbf{X}\}$, then $(\mathbf{X}, \mathcal{A})$ is obviously a block design, and it is said to be an *obvious* design. Also, if $\mathcal{A}$ is a set of the k-subsets of $\mathbf{X}$, then $(\mathbf{X}, \mathcal{A})$ is obviously a block design, and it is said to be a *complete* design. If a block design is neither obvious nor complete, it is called a *balanced incomplete block design* (*BIBD*) and denoted as (v, b, r, k, λ)-BIBD. The five parameters are not all independent; the basic two equations are

$$bk = vr, \tag{2.19}$$

$$\lambda(v-1) = r(k-1). \tag{2.20}$$

Thus, it is not uncommon to write a BIBD as a (v, k, λ)-BIBD.

The most basic necessary condition for the existence of a BIBD known as *Fischer's inequality*, named after the statistician Ronald Fisher, states that a (v, b, r, k, λ)-BIBD exists only if $b \geqslant v$ (or equivalently, if $r \geqslant k$). A BIBD with $b = v$ (or equivalently, $r = k$) is called a *symmetric* BIBD. The parameters of a symmetric design satisfy

$$\lambda(v-1) = k(k-1). \tag{2.21}$$

Finite projective planes are symmetric BIBD with $\lambda = 1$. From Eq. 2.21, finite projective planes satisfy

$$v - 1 = k(k-1). \tag{2.22}$$

Since $r = k$ holds by the definition of a symmetric BIBD, the order n of a finite projective plane is equal to $k - 1$. From Eq. 2.22, we obtain $v = (n+1)n + 1 = n^2 + n + 1$ points in a finite projective plane of order n. Thus, a finite projective plane of order n is a $(n^2 + n + 1, n + 1, 1)$-BIBD.

From a (v, b, r, k, λ)-BIBD, we can construct a graph called an *incidence graph* (a.k.a. *Levi graph*), which is a bipartite graph $G = (V_1, V_2, E)$ where

- V_1 is a set of points,
- V_2 is a set of blocks, and
- E is a set of edges that connect a point p_i and a block B_j if and only if p_i occurs in B_j.

The incidence graph of a BIBD will be applied to a network topology in Chapter 4.

Chapter 3

Low-Latency Interconnection Networks with Single-Port Hosts

3.1 Overview

In this chapter, 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 analyzing 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?*

First, Section 3.2 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 3.3-3.4 present host-switch graphs with low h-ASPL; we take deterministic and heuristic approaches in each section. 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 3.5, we practically compare proposed network topologies with existing ones in terms of performance, topological properties, power consumption, and cost breakdowns. Section 3.6 reviews related work, and finally, we conclude the chapter in Section 3.7.

3.2 Introduction of a Host-Switch Graph

3.2.1 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*),

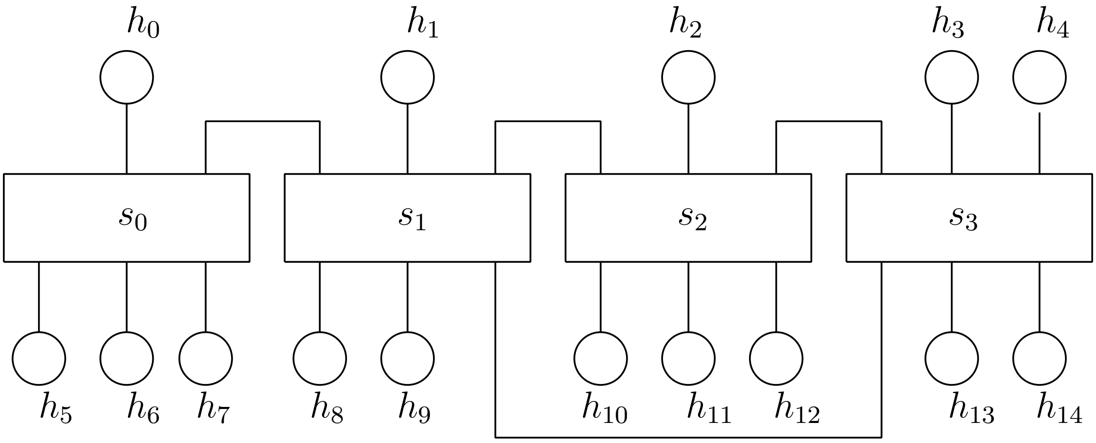


Figure 3.1: An example of a host-switch graph ($n = 15, m = 4, r = 6$).

- $S = \{s_0, s_1, \ldots, s_{m-1}\}$ is a set of m elements called *switch vertices* (or simply *switches*), and
- $E \subset \{\{s_i, s_j\} \mid s_i, s_j \in S\} \cup \{\{h_i, s_j\} \mid (h_i \in H) \wedge (s_j \in S)\}$ is a set of unordered pairs of connected vertices 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. 3.1 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 3.1 (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. 3.1 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

$$D(G) := \max\{\ell(h_i, h_j) \mid 0 \leqslant i < j < n\}.$$

The *host-to-host average shortest path length (h-ASPL)* $A(G)$ is defined as

$$A(G) := \sum_{0 \leqslant i < j < n} \ell(h_i, h_j) / \binom{n}{2}.$$

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.

3.2.2 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 $D_{h_s}(G)$ respectively denote a single-source h-ASPL from h_s and a single-source diameter from h_s, as follows:

$$\begin{aligned} A_{h_s}(G) &:= \sum_{0 \leqslant i < n, i \neq s} \ell(h_s, h_i)/(n-1), \\ D_{h_s}(G) &:= \max\{\ell(h_s, h_i) \mid 0 \leqslant i < n \text{ and } i \neq s\}. \end{aligned}$$

Using these notations, we obtain the upper bound on the order, as follows:

Theorem 3.1 (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

$$\begin{cases} |H_0| = 1, & \text{if } i = 0 \\ |S_1| = 1, & \text{if } i = 1 \\ |S_{i-1}|\,(r-1)\,. & \text{if } i > 1 \end{cases} \tag{3.1}$$

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 3.1:

Corollary 3.1 (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 3.2 (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 3.2, 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

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

3.3 Deterministic Construction of Host-Switch Graphs

3.3.1 Radix/Diameter Problem

We discuss host-switch graphs that can deterministically be constructed. Since the relationship between the diameter and the order is easy to analyze, we discuss the *radix/diameter problem* (*RDP*), which is similar to a classical problem called the degree/diameter problem [48, 86]. The RDP is defined as follows:

Problem 3.1 (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 3.1.

3.3.2 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 3.3 (Upper bound on the order of a host switch graph of diameter 2)**.** *A host-switch graphs of diameter 2 can connect at most r hosts.*

3.3.3 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 3.2 (Condition for constructing a host-switch graph of diameter 3)**.** *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 3.4 (Upper bound on the order of a host-switch graph of diameter 3)**.** *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.*

3.3.4 Host-Switch Graphs of Diameter 4

3.3.4.1 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*.

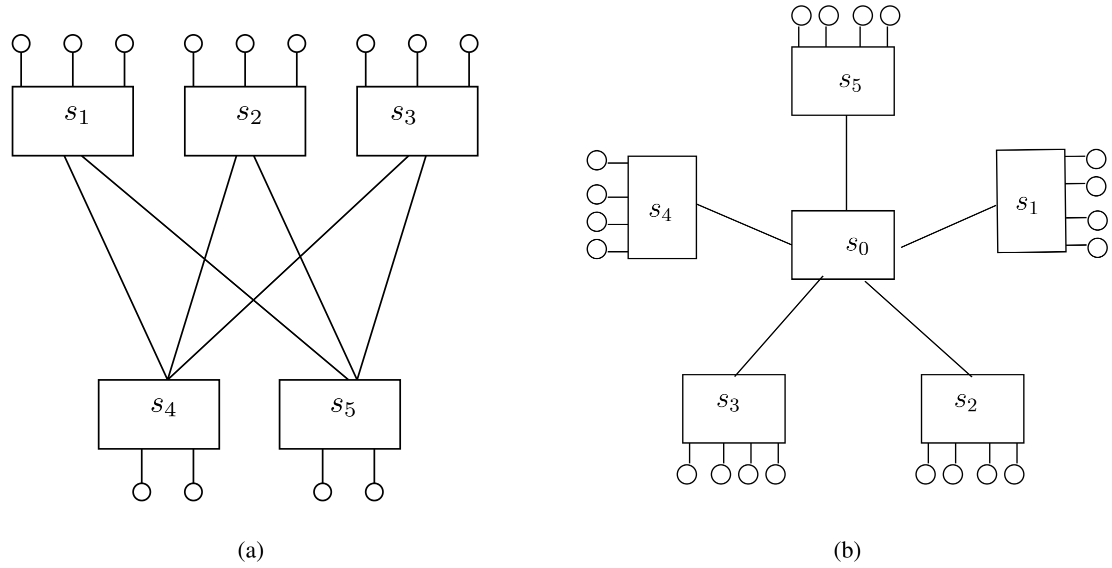


Figure 3.2: Examples of a biclique host-switch graph with $r = 5$; (a) $\{m_1, m_2\} = \{3, 2\}$, $n = 13$ and (b) star host-switch graph with $n = 20$.

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. 3.2a 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 3.3 (Upper bound on the switch order of a biclique host-switch graph)**.** *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

$$m_1 (r - m_2) + m_2 (r - m_1) = r (m_1 + m_2) - 2m_1 m_2. \qquad (3.2)$$

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 $r^2 - m_2 r$. This value is maximized when $m_2 = 1$. Consequently, $n_{\max}$ is at most $r(r - 1)$ in this case.

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.

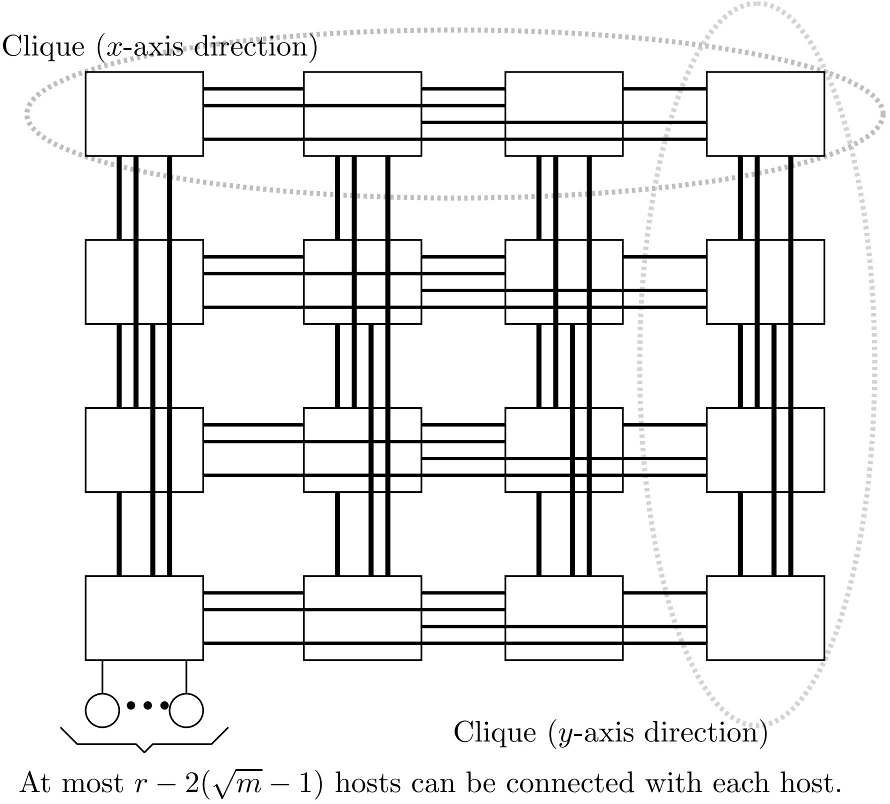

Clique (x-axis direction)
Clique (y-axis direction)
At most $r - 2(\sqrt{m} - 1)$ hosts can be connected with each host.

Figure 3.3: An example of an XY-clique host-switch graph.

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

Theorem 3.5 (Upper bound on the order of a biclique host-switch graph). *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 [98]. In Fig. 3.2b we illustrate an example of a star host-switch graph.

3.3.4.2 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. 3.3 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 3.4 (Conditions for constructing an XY-clique host-switch graph). *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 3.6 (Upper bound on the order of an XY-clique host-switch graph)**.** *An XY-clique host-switch graph can have at most*

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

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 3.7 (The order of an XY-clique and star host-switch graphs)**.** *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* [70].

3.3.4.3 Polarity Host-Switch Graph

A *polarity graph* [17] (a.k.a. *Brown's construction* or *Brown graph* [29]) 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 3.5 (Upper bound on the order of a polarity host-switch graph)**.** *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

$$r = (3q^2 + 4q + 1)/(2q + 1). \tag{3.3}$$

Substituting this value of r to $r(q^2+q+1)-q(q+1)^2$, $n_{\max}$ becomes

$$(q^4+2q^3+4q^2+4q+1)/(2q+1). \tag{3.4}$$

Also, from Eq. 3.3 , $n_{\max}$ is maximized when $q \sim 2r/3$. Substituting this value of q to Eq. 3.4, we obtain $n_{\max} \sim 4r^3/27$, which is better than the upper bound on the order of an XY-clique host-switch graph (see Theorem 3.6). Thus, a polarity host-switch graph can potentially connect more hosts than a star host-switch graph and an XY-clique host-switch graph.

3.3.5 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 A). 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 3.2b (≈ 3.68) is higher than the h-ASPL of a biclique host-switch graph in Fig 3.2a (≈ 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.

3.4 Heuristic Construction of Host-Switch Graphs

3.4.1 Order/Radix Problem

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.

First, let us formulate a problem to solve. Unlike the deterministic approach described in Section 3.3, a heuristic one enables us to optimize h-ASPL directly, and hence the problem is below.

Problem 3.2 (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 [90, 107] that searches an undirected regular graph with low diameter/ASPL. We adopt simulated annealing (SA) [73] to escape from a local solution.

3.4.2 Minimizing h-ASPL

3.4.2.1 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.

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 $\mathscr{O}(|V||E|)$ running time in total. The h-ASPL is given by $\sum_{0\leqslant i<j<m}(\ell(s_i, s_j) + 2) \cdot w_i w_j / \binom{n}{2}$, 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 $\mathscr{O}(m^2 r - mn)$ running time. Note that the order n does not increase the time complexity. On the contrary, n decreases it. Also, it reduces to $\mathscr{O}(m^2)$ if n takes the upper bound given by Lemma 3.1, 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 [33] for large m or dense graphs.

3.4.2.2 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. 3.4), 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 n/m, the lower bound on the h-ASPL of a k-regular host-switch graph is obtained by the Moore bound (see Eq. 2.10), the lower bound on the ASPL of a graph with N vertices and maximum degree K, say $M(N, K)$, as follows:

$$\begin{aligned} A(G) &\geqslant \frac{(n/m)^2\binom{m}{2}(M(m, r - n/m) + 2) + 2\binom{n/m}{2}m}{\binom{n}{2}} \\ &= \frac{M(m, r - n/m)(mn - n)}{mn - m} + 2. \end{aligned} \tag{3.5}$$

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

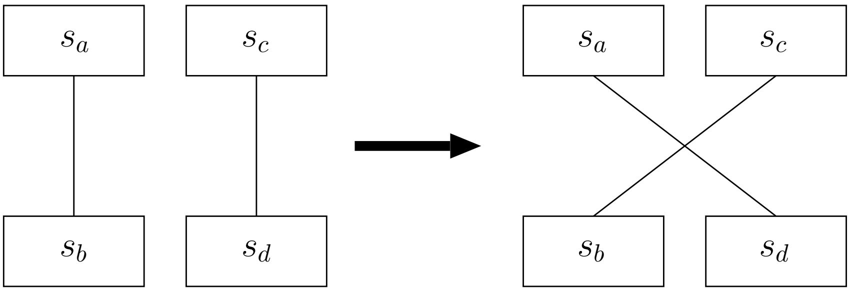


Figure 3.4: Swap operation which changes endpoints of two switch-switch edges.

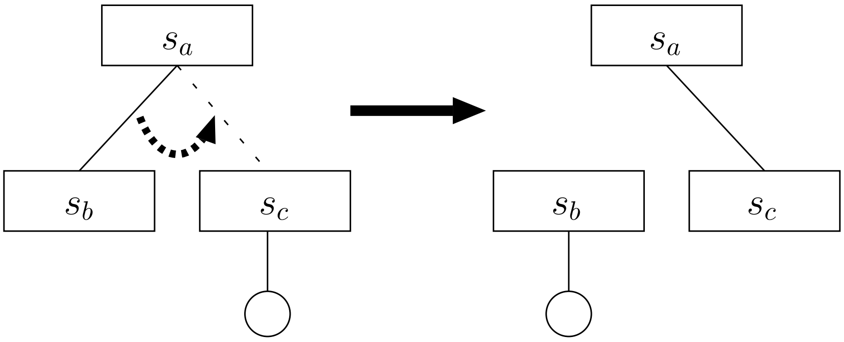


Figure 3.5: Swing operation which changes endpoints of a switch-switch edge and a host-switch one.

3.4.2.3 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. 3.5). 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. 3.6). 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

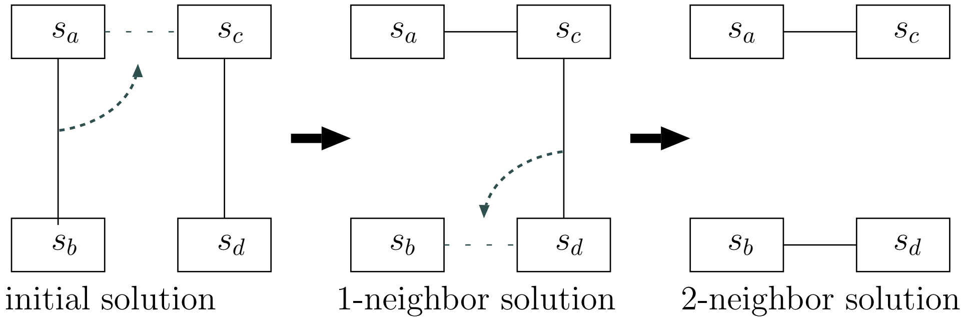


Figure 3.6: 2-neighbor swing operation (hosts are omitted for simplicity).

initial solution holds.

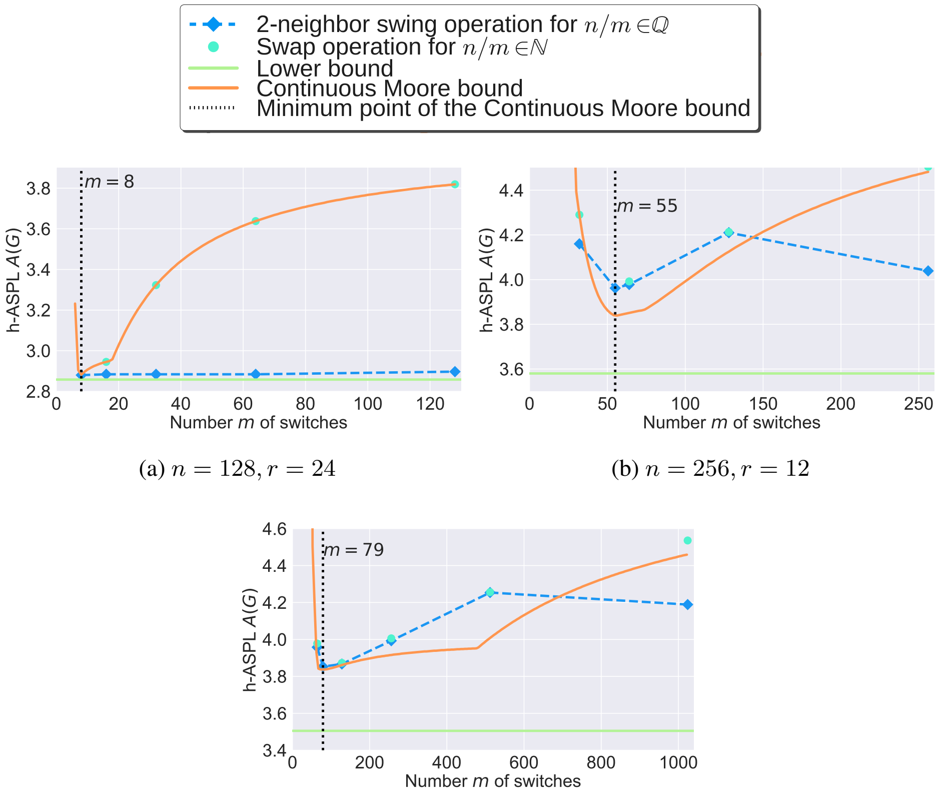
Consequently, this operation contains both of the swap operation (if the 2-neighbor solution is accepted) and the swing operation (if the 1-neighbor solution is accepted).

3.4.2.4 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 3.2 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. 3.7. We pick up the typical results, and other cases are similar to either of the three results. Here, the Moore bound is calculated by (3.5); however, n/m must be an integer, and 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. 3.9 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. 3.7a), because the switches can constitute a clique only in this case as described in Section 3.3. In other words, n can be less than $m(r - m + 1)$ only in this case (see Lemma 3.2). Thus, the h-ASPL tends to be large in other cases. Also, in this case, the h-ASPL is close to the lower bound derived by Theorem 3.2, 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. 3.7, 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. 3.8 we show the distribution of the number of connected hosts of a switch, which we call the *host distribution*. Interestingly, the



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

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

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

Figure 3.7: Relationship between h-ASPL and the number of switches.

obtained graph includes switches that have different number of hosts. This corresponds to neither conventional direct nor indirect networks.

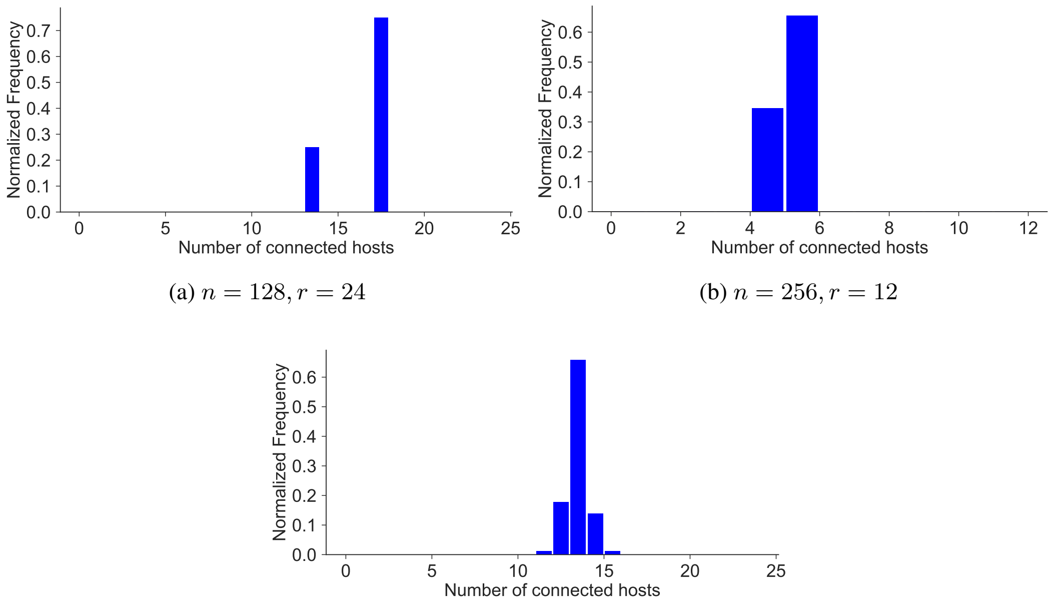
Other phenomenon of interest is that, when $m < m_{\text{opt}}$ or $m \gg m_{\text{opt}}$, the h-ASPL of a regular host-switch graph significantly exceeds A_{opt} as compared with the h-ASPL of a non-regular host-switch graph. Let us discuss each case.

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. 3.7c, the host distribution is similar to Fig. 3.10, 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 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



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

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

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

Figure 3.8: Host distribution when $m = m_{\text{opt}}$.

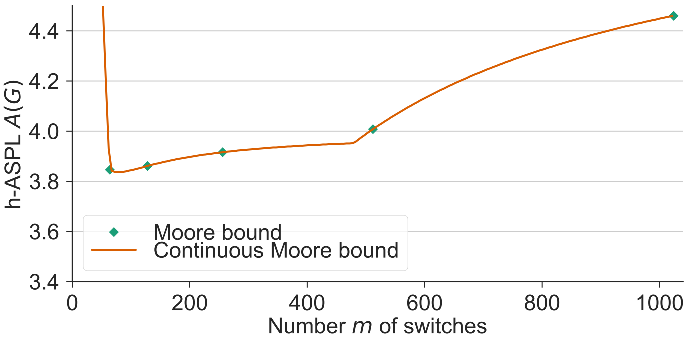


Figure 3.9: Comparison between the Moore bound and the continuous Moore bound.

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 3.1 (Relationship between the h-ASPL and the continuous Moore bound)**.** *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.

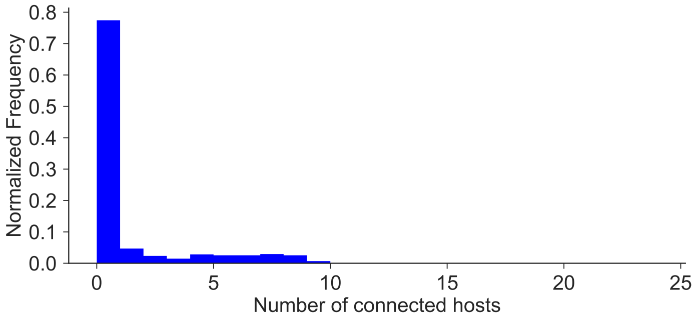


Figure 3.10: Host distribution of a host-switch graph with unused switches when $(n, m, r) = (1024, 1024, 24)$.

3.4.3 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.

3.4.3.1 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 [47]. 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 [16, 42, 91] and its bounds based on spectral graph theory [23] are studied, its calculation for arbitrary graph is NP-complete [58]. We hence compute it approximately by using a graph partitioning software called hMETIS [65], which is a family of METIS [66] 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 $\lfloor n/2 \rfloor$.

3.4.3.2 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

[1] Note that this definition is more practical than that in [124], 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.

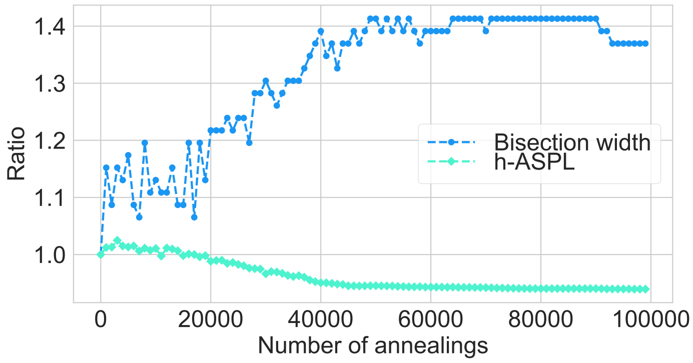


Figure 3.11: Changes of h-ASPL and BiW during optimization.

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.

3.4.3.3 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 [46] 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. 3.11 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:

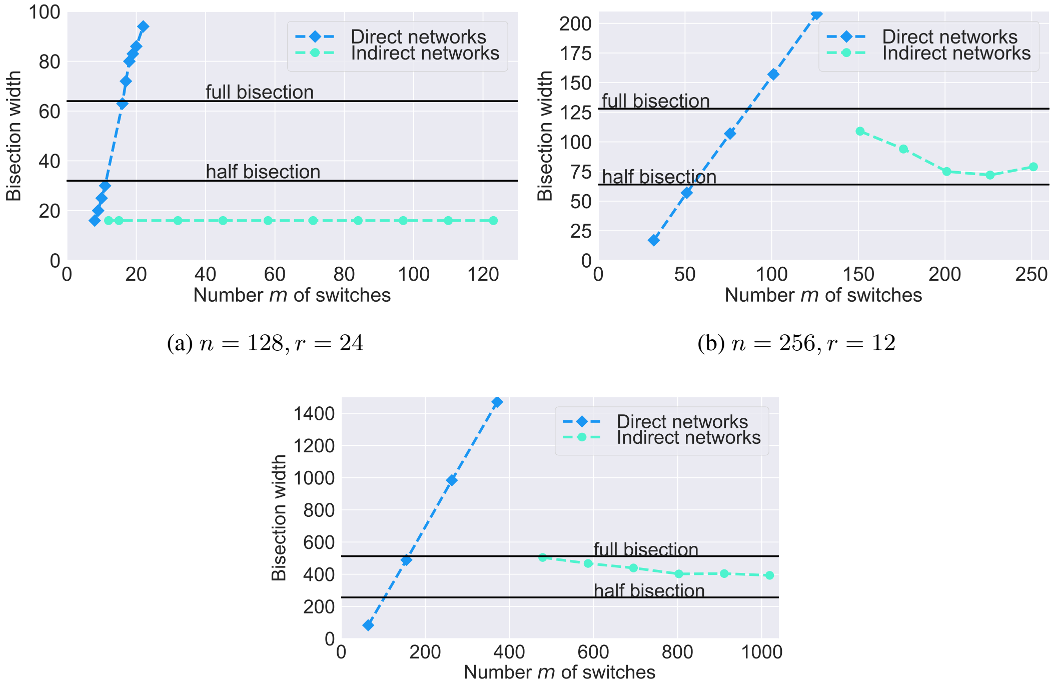
Observation 3.2 (Relationship between the h-ASPL and the BiW)**.** *In a host-switch graph with fixed n, m, and r, reducing the h-ASPL yields increasing the BiW.*

Thus our method described in Section 3.4.2 can be used also for maximizing the BiW.

3.4.3.4 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. 3.12 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



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

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

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

Figure 3.12: Relationship between the BiW and the number of switches.

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 3.3 (Linear relation between the number of switches and the BiW)**.** *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.

3.4.4 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 [90]).

- 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 3.5 with existing topologies.

3.5 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 3.3, previously proposed topologies, and existing topologies applied to supercomputers ranked in TOP500 and demonstrate the advantage of our proposed topologies.

3.5.1 Previously proposed topologies

3.5.1.1 WK-recursive networks

The WK-recursive network [118] $\mathrm{WK}(K, L)$ 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

$$m = K^L. \tag{3.6}$$

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:

$$r \geqslant K, \tag{3.7}$$

$$n \leqslant K^L(r-K+1)-(L-1)(K-1). \tag{3.8}$$

3.5.1.2 Recursive dual-net

The recursive dual-net $\mathrm{RDP}^k(B)$ [80] 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 [80], 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:

$$m = (2m_B)^{2^k}/2, \tag{3.9}$$

$$n \leqslant (r-k-d_B)\cdot(2m_B)^{2^k}/2, \tag{3.10}$$

$$r > d_B + k. \tag{3.11}$$

3.5.2 Existing topologies

There are many existing topologies for practical interconnection networks. Among them, we pick up three typical topologies: the torus [37], the dragonfly [71], and the fat-tree [12]. They are applied to supercomputers ranked in TOP500 for June 2017 [3]; for example, Titan [25] and Sequoia [88] use the torus, Cori [15] and Piz Daint [13] use the dragonfly, and Tianhe-2 [81] 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.

3.5.2.1 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 $j \neq i$).

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

$$m = K^N. \tag{3.12}$$

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

$$n \leqslant (r - 2N) \cdot K^N, \tag{3.13}$$

$$r > 2N. \tag{3.14}$$

3.5.2.2 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 [71], 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

$$m = ag = \frac{a^3}{2} + a. \tag{3.15}$$

The radix r of a dragonfly host-switch graph is

$$r = (a - 1) + h + p = 2a - 1. \tag{3.16}$$

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

$$n \leqslant mp = \frac{a^4}{4} + \frac{a^2}{2}. \tag{3.17}$$

3.5.2.3 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* [12]. 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

$$m = 5K^2/4. \tag{3.18}$$

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:

$$n \leqslant K^3/4. \tag{3.19}$$

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

$$r = K. \tag{3.20}$$

Table 3.1: Summary of fundamental properties of host-switch graphs.

Graph	**Section**	**Parameters**	**Order** n	**#Switches** m	**Radix** r	**Diameter**
Clique	3.3	–	$m(r-m+1)$	$\leqslant r+1$	given	3
Star	3.3	–	$r(r-1)$	$r+1$	given	4
XY-Clique	3.3	–	$m(r-2\sqrt{m}+2)$	$< (r+2)^2/4$	given	4
Polarity	3.3	q	$r(q^2+q+1)-q(q+1)^2$	q^2+q+1	$> q+1$	4
Randomly-optimized	3.4	–	$(r-1)^{D(G)-1}+1$	variable	given	$\geqslant \lceil \log_{r-1}(n-1) \rceil + 1$
WK-recursive	3.5.1	K, L	$K^L(r-K+1)-(L-1)(K-1)$	K^L	$\geqslant K$	2^L+1
Recursive dual-net	3.5.1	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	3.5.2	K, N	$(r-2N)\cdot K^N$	K^N	$> 2N$	$\lfloor K/2 \rfloor \cdot N+2$
Dragonfly	3.5.2	a	$a^4/4+a^2/2$	$a^3/2+a$	$2a-1$	5
Fat-tree	3.5.2	K	$K^3/4$	$5K^2/4$	K	6

3.5.3 Fundamental Properties

Table 3.1 summarizes the fundamental properties of host-switch graphs presented in Sections 3.3, 3.4, 3.5.1, and 3.5.2. We have considered graphs presented in Sections 3.3 and 3.4, so let us now compare them with the graphs presented in Section 3.5.1 and 3.5.2. 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 $\mathscr{O}(2^L)$. 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 $\mathrm{WK}(K, 2)$ and the dragonfly.

One might notice that $\mathrm{WK}(K, 2)$ and the dragonfly are quite similar; technically, $\mathrm{WK}(K, 2)$ is a dragonfly such that a is equal to g, h is equal to 0 or 1, and p is variable (cf. Section 3.5.2.2). Let us consider $\mathrm{WK}(K, 2)$ with the parameters of the dragonfly for comparison. Since K corresponds to a, the number m of switches is

$$m = a^2. \tag{3.21}$$

The order n must satisfy:

$$n \leqslant a^2(r - a + 1) - (a - 1) = -a^3 + a^2(r + 1) - a + 1. \tag{3.22}$$

When we assume $r = 2a - 1$ as with the dragonfly, (3.22) reduces to

$$n \leqslant a^3 - a + 1. \tag{3.23}$$

As a result, the dragonfly can connect more hosts than $\mathrm{WK}(K, 2)$ 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 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.

3.5.4 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.

3.5.4.1 Evaluation of Performance and Topological Properties

The performance is evaluated by SimGrid discrete event simulator (v3.15) [32]. 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) [26] and simulate MPI implementation of NAS parallel benchmarks (v3.3.1, Class A for IS and FT and Class B for the others) [8].

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. Afterward, hosts are sequentially connected to switches in depth-first order by backtracking.

Table 3.2 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 (3.12)–(3.14) we set K and r to 3 and 15, respectively. Consequently the torus satisfies $n \leqslant 1215$, $m = 243$, and $r = 15$. From (3.15)–(3.17) 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 (3.18)–(3.20) 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 3.2, 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.

Table 3.2: 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 3.2**	**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%)

3.5.4.2 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 [21]. 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.

3.5.5 Results and Discussion

3.5.5.1 Comparison with 5D Torus

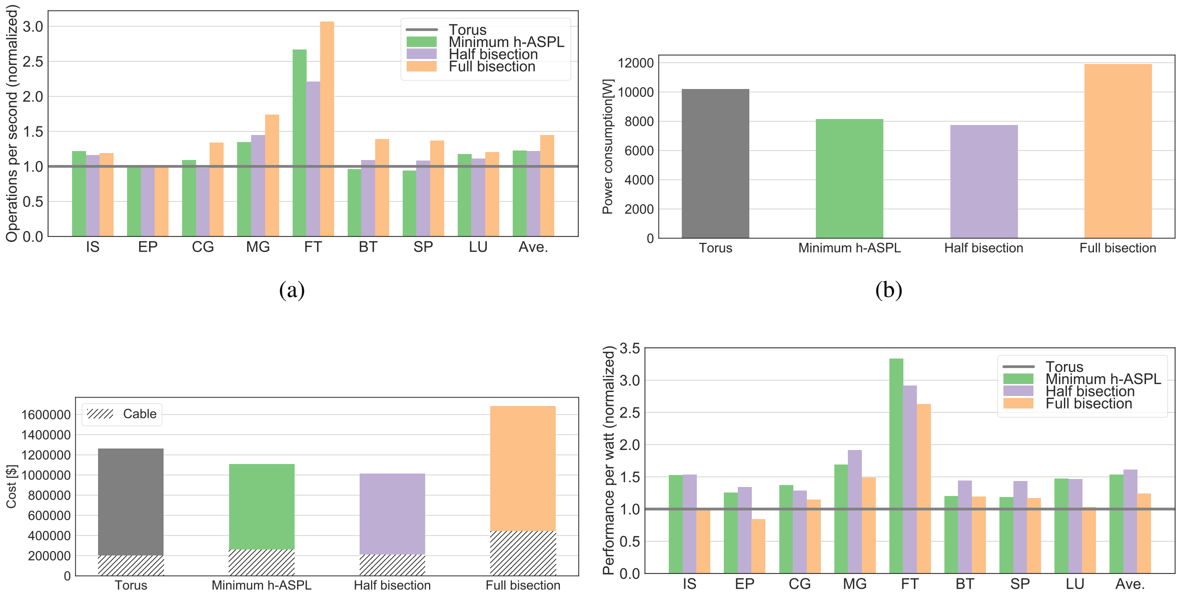
From Table 3.2, the h-ASPL of the torus (≈ 5.34) is much higher than the continuous Moore bound (≈ 4.47). On the other hand, our topologies have low h-ASPLs close to the Moore bound. Also in terms of the BiW, all of our topologies are better than the torus. It is interesting to note that our topology with the minimum h-ASPL and the half-bisection one provide similar topological properties.

In Fig. 3.13a 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), 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. 3.13b 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. 3.13c 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. 3.13d 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,



(a)

(b)

(c)

(d)

Figure 3.13: 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.

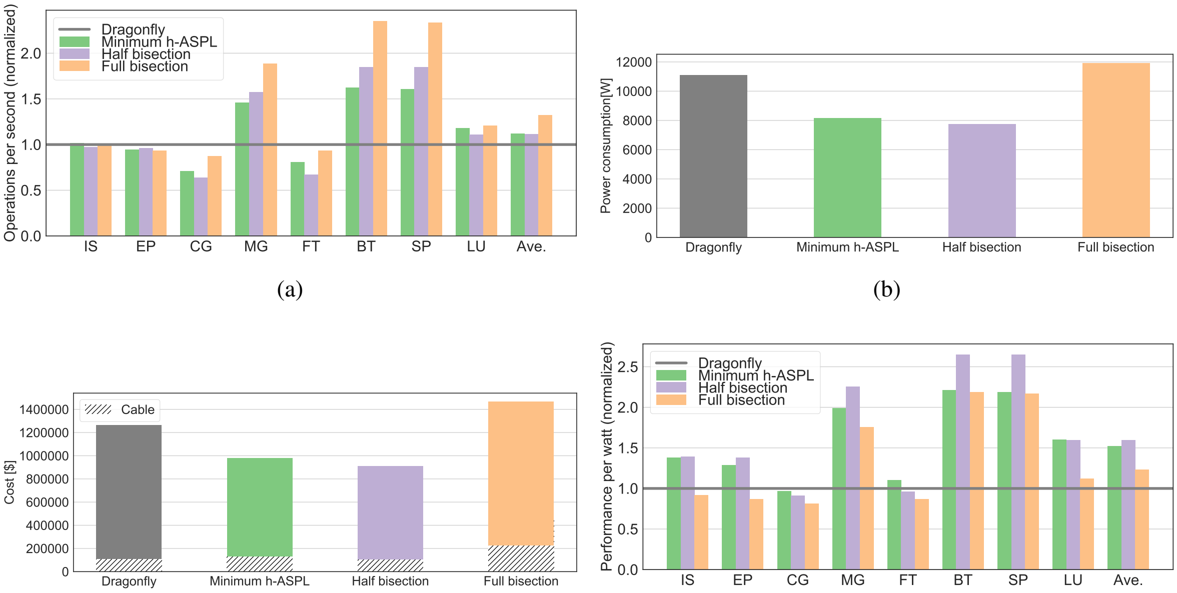
our half-bisection topology provides the best improvement (61% on average). On the other hand, our full-bisection topology improves slightly since it consumes large power.

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.

3.5.5.2 Comparison with Dragonfly

From Table 3.2, 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 3.14a. 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



(a)

(b)

(c)

(d)

Figure 3.14: 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.

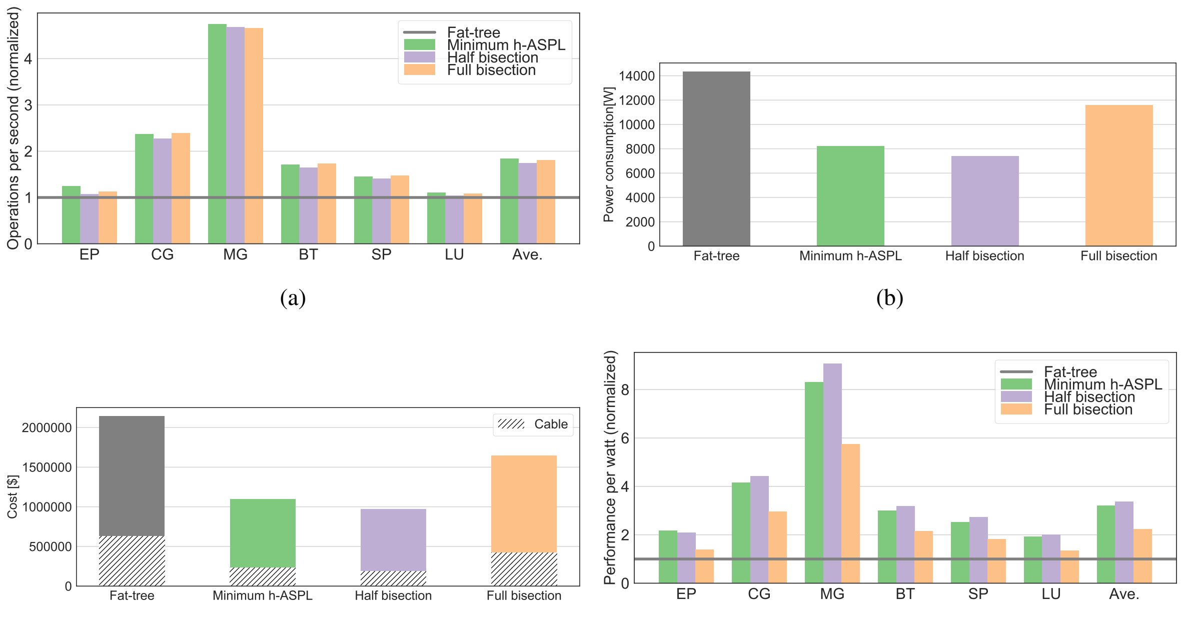
performing poorer on the proposed networks than the dragonfly. This is because the EP benchmark requires few communications, and hence the h-ASPL has little effect on the performance. But rather, the application mapping affects the performance. Note that the performance of the EP is hard to change even if the network changes.

In Fig. 3.14b 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. 3.14c 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. 3.14d 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. 3.13d.

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



(a)

(b)

(c)

(d)

Figure 3.15: 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.

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.

3.5.5.3 Comparison with Fat-tree

From Table 3.2, 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. 3.15a 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. 3.13a , 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. 3.15b 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 topologies 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. 3.15c we show the results of the cost comparison. Unlike the torus and the dragonfly, the fat-tree requires 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. 3.15d 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. 3.13d and 3.14d), 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.

3.5.6 Practical Feasibility and Limitations

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

3.5.6.1 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 [45,69,101], using virtual channels [112], and forwarding every flit in the deadlocked ring at the same time [97]. 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 (a.k.a. forwarding 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 [105] or InfiniBand [1]. For all these systems, the routing table size is thus also a scalability limitation regardless of the network topology.

3.5.6.2 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.

3.5.6.3 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 3.1) 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.

3.6 Related Work

Curtis *et al.* proposed a method for designing high-performance data center interconnection networks with various switches and cables [36]. This method, called *REWIRE*, optimizes the performance (determined by the diameter and the bisection bandwidth) within the required costs. Since they use various switches such that the number of ports and the bandwidth are different, it is difficult to predict the switching latency and costs.

3.7 Summary

In this chapter, 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 single-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 efficiently 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.

Chapter 4

Low-Latency Interconnection Networks with Multi-Port Hosts

4.1 Overview

Thus far we have studied interconnection networks with single-port hosts. However, we can increase not only the number of ports of a switch but also the number of ports of a host. This paper thus studies the impact of increasing the number of ports of a host. Multi-port hosts are typically used for link aggregation between a host and a switch, which increases available bandwidth [120]. Another purpose of using multi-port hosts is increasing fault-tolerance by connecting multiple independent networks [122]. We label the networks above link aggregation (LA) and network duplication (ND), respectively. In addition to these purposes, we propose two methods for reducing the hop count, and consequently we can reduce end-to-end latencies without increasing the number of ports of a switch. One of our proposed methods is the permutation of host-switch mapping that can be applied to LA and ND, and we label the obtained network p-LA and p-ND, respectively. The other proposed method is the application of a finite projective plane, which we label PP. The use of multi-port hosts is agnostic to the switch topology, and thus multi-port hosts can efficiently be applied existing topologies, including both random and non-random topologies.

4.2 Preliminary

4.2.1 Extension of a Host-Switch Graph

We extend a host-switch graph described in Section 3.2 so that a host vertex can be connected with multiple edges as follows. A *host-switch graph* is a 3-tuple $G = (H, S, E)$ with integer parameters $n \geqslant 3$, $m \geqslant 1$, $r \geqslant 3$, and $p \geqslant 1$ 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

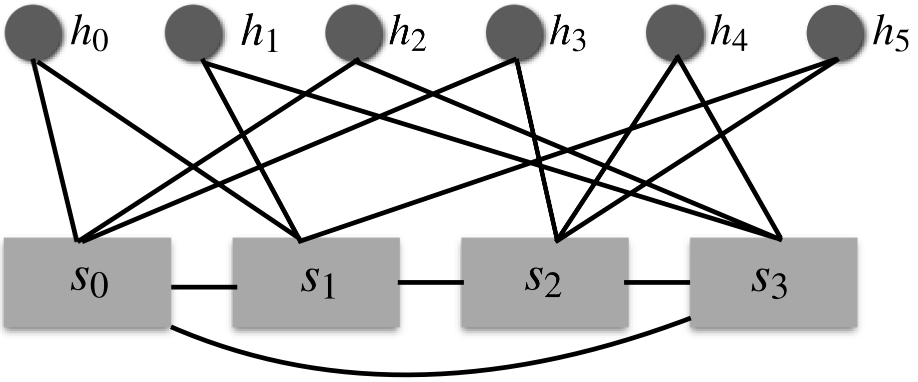


Figure 4.1: An example of a host-switch graph with 6 2-port hosts and 4 5-port hosts.

- $E \subset \{\{s_i, s_j\} \mid s_i, s_j \in S\} \cup \{\{h_i, s_j\} \mid (h_i \in H) \wedge (s_j \in S)\}$ is a set of elements called *edges*. Edges $\{s_i, s_j\}$ and $\{h_i, s_j\}$ are specifically called a *switch-link* and a *host-link*, respectively.

The number n of hosts is called the $order$ of G. Each switch is connected with at most r edges; the number r of required ports per switch is called the $radix$ of G. Each host is connected with at most p edges; the upper bound p of the number of edges connected to a host is called the *host-degree* of G. Actually, in the previous chapter, we have studied a host-switch graph with host-degree $p = 1$; in this chapter, we further study cases when p is greater than one.

In Fig. 4.1 we illustrate an example of a host-switch graph with $n = 6$, $m = 4$, $r = 5$, and $p = 2$. Throughout this paper, a circle and a rectangle represent respectively a host and a switch. A *path* between hosts in a host-switch graph is a sequence of edges that includes exactly two host-links; in other words, there exist no intermediate hosts. 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, $\ell(h_0, h_4)$ in Fig. 4.1 is three because the shortest path between them is (h_0, s_0, s_3, h_4). Note that there exist longer alternative paths between them.

Using $\ell(h_i, h_j)$, let us define two topological properties. The *diameter* is the maximum length of the shortest path defined as

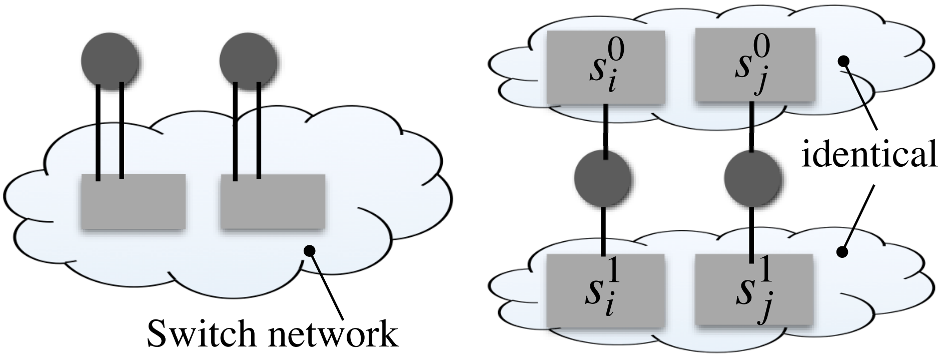
$$\max\{\ell(h_i, h_j) \mid 0 \leqslant i < j < n\}.$$

The *host-to-host average shortest path length* (*h-ASPL*) is defined as

$$\sum_{0 \leqslant i < j < n} \ell(h_i, h_j) / \binom{n}{2}.$$

4.2.2 Assumptions

Interconnection networks that we study in this paper satisfy the following design assumptions. First, networks can use routing tables for routing packets, and hence we can use arbitrary topologies



(a) (b)

Figure 4.2: Conventional interconnection networks with multi-port hosts when $p = 2$. (a) Link Aggregation (LA). (b) Network Duplication (ND).

including random and non-random ones. Costs of routing tables are discussed in Section 4.5.5. Dead-lock can be avoided by previous methods using virtual channels [69, 112]. Second, the hosts never forward packets; in other words, there exist no intermediate hosts along the path between a source host and a destination host. This assumption avoids massive overheads in a host. We review cases that do not satisfy the second assumption in Section 4.6.2.

4.3 Conventional Methods Revisited

We now revisit conventional networks with multi-port hosts. Existing networks are classified into two classes which we label:

1. Link Aggregation (LA); and
2. Network Duplication (ND).

Fig. 4.2 illustrates conceptual diagrams of the two networks above, whose details are described below.

4.3.1 Link Aggregation (LA)

A link aggregation (LA), shown in Fig. 4.2a, is one of the conventional uses of multi-port hosts. It bundles multiple cables between a host-switch pair to increase bandwidths of host-links. Modern commodity switches, network interfaces, and communication libraries usually support LA such as IEEE 802.3ad. The reason for using LA is that an injection or reception host-link often become a bottleneck in an interconnection network for communication-intensive workloads. Some HPC platforms are thus provisioned with relatively high host-link bandwidth; for instance, in the case of the IBM BlueGene/L [11], injection and reception bandwidths of host-links are respectively 612.5 MB/s and 1050 MB/s, whereas bandwidths of switch-links are 350 MB/s. We formally define networks using LA as follows:

Definition 4.1 (LA)**.** *A network using* LA *is represented as* $\mathrm{LA}(G, p)$ *for* $p > 1$*, where* G *is a host-switch graph of host-degree 1.* $\mathrm{LA}(G, p)$ *is a host-switch graph of host-degree* p *obtained by multiplying every host-link in* G *by* $p - 1$ *times.*

As a result, the radix of $\mathrm{LA}(G, p)$ becomes more than the radix of G. Clearly, the h-ASPL and the diameter of $\mathrm{LA}(G, p)$ is equal to those of G, while other topological properties such as BiW may be different (we show this in Section 4.5).

4.3.2 Network Duplication (ND)

Another conventional use of multi-port hosts is for supporting different interconnection networks in a single system. For example, the SGI Altrix3000 platform comprises two identical topologies [122], each host having two network interfaces. Two use-cases of different-device networks have been proposed for HPC platforms: optical circuit switching networks for long bulk data transfer and lower-bandwidth electric packet switching [18]. We assume each network topology, called a *switch network*, is identical for simplicity and call this class *network duplication* (*ND*). It is depicted in Fig. 4.2b. We formally define networks using ND as follows:

Definition 4.2 (ND)**.** *A network using* ND *is represented as* $\mathrm{ND}(G, p)$ *for* $p > 1$*, where* $G = (H, S, E)$ *is a host-switch graph.* $\mathrm{ND}(G, p)$ *is a host-switch graph obtained as follows. Make* p *copies of* S *and* E*, and denote them* S_i *and* E_i*, respectively, for* $0 \leqslant i \leqslant p - 1$ *where*

- $S_i = \{s_0^i, s_1^i, \ldots, s_{m-1}^i\}$*, and*
- $E_i = \{\{s_j^i, s_k^i\} \mid \{s_j, s_k\} \in E)\} \cup$
  $\{\{h_j, s_k^i\} \mid (h_j \in H) \wedge (\{h_j, s_k\} \in E)\}$.

In that case, $\mathrm{ND}(G, p)$ *is* (H', S', E') *such that*

- $H' = H$*,*
- $S' = S_0 \cup S_1 \cup \cdots \cup S_{p-1}$*, and*
- $E' = E_0 \cup E_1 \cup \cdots \cup E_{p-1}$.

As with LA, the h-ASPL and the diameter of $\mathrm{ND}(G, p)$ is equal to those of G, while other topological properties such as BiW may be different (we show this in Section 4.5).

4.4 Proposed Methods

Thus far, we review two traditional networks with multi-port hosts that cannot reduce the diameter and the h-ASPL. From here, we propose two methods for reducing them: permutation of host-switch mapping and using small switch networks.

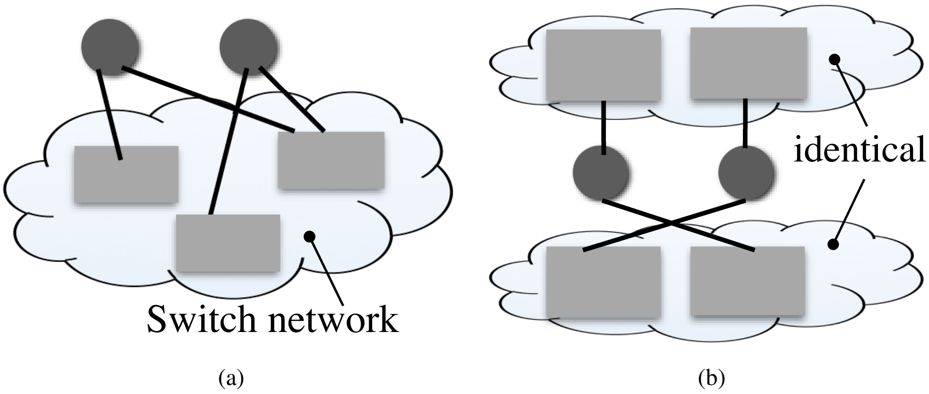


Figure 4.3: Proposed methods when $p = 2$. (a) Permutation for Link Aggregation (p-LA). (b) Permutation for Network Duplication (p-ND).

4.4.1 Permutation of Host-Switch Mapping

Let us $(a_0, a_1, \ldots, a_{np-1})$ denote a sequence of np switches such that p switches $a_{ip}, a_{ip+1}, \ldots, a_{ip+p-1}$ are connected to the host h_i ($0 \leqslant i \leqslant n-1$) and call this sequence *host-switch mapping*. For example, the host-switch mapping of LA when $p = 3$ becomes $(s_i, s_i, s_i, s_j, s_j, s_j, \ldots)$, and that of ND when $p = 3$ becomes $(s_i^0, s_i^1, s_i^2, s_j^0, s_j^1, s_j^2, \ldots)$. We can design a new host-switch graph by changing the host-switch connection of an existing host-switch graph. It should be clear that a permutation of the host-switch mapping can be used to define the host-switch connection of the new host-switch graph. According to the host-switch mapping, the diameter and the h-ASPL are capable of decreasing. Figs. 4.3a and 4.3b illustrate permutated LA and ND, respectively.

Since the optimal permutation for reducing the h-ASPL and the diameter is not obvious, we use simulated annealing (SA) to optimize the permutation. In SA, two host-switch links are randomly selected, and their end-points are swapped. During optimization, we calculate the h-ASPL as follows.
Step 1: Compute the shortest path lengths from every switch of a host-switch graph G by using the breadth-first search (BFS).
Step 2: Using the results of Step 1, compute the h-ASPL of a host-switch graph by comparing all the possible shortest paths between two hosts.
We need Step 1 only once and then repeat Step 2.

The time complexity in Step 1 depends on that of BFS, i.e., $\mathcal{O}(|V||E|)$ for a graph (V, E). In the case of a host-switch graph, $|V|$ and $|E|$ correspond to the number of switches in a switch network and the number of switch-links in a switch network, respectively. Let us consider the case of LA. Since there exists only one switch network, $|V|$ is equal to m. From the number of all the edges $(mr/2)$ and the number of host-links $(np/2)$, $|E|$ is equal to $(mr - np)/2$. Thus, in the case of LA, the time complexity in Step 1 is $\mathcal{O}(m^2r - mnp)$. Next, let us consider the case of ND. Since there exit p switch network, $|V|$ is equal to m/p. From the number of all the edges $(mr/2p)$ and the number of host-links ($n/2$), $|E|$ is equal to $(mr/2p - n/2)$. Thus, in the case of ND, the time complexity in Step 1 is $\mathscr{O}((m^2r - mnp)/p^2)$.

The time complexity in Step 2 depends on the number of possible shortest paths between two hosts; it is p^2 and p for LA and ND, respectively. Thus, it takes $\mathscr{O}(n^2p^2)$ and $\mathscr{O}(n^2p)$ running time for LA and ND, respectively.

4.4.2 Using Small Switch Networks

An alternative approach for reducing the diameter and the h-ASPL is using many identical switch networks whose order (the number of connected hosts) is as less as possible—we show that the minimum value can be less than n. Since every switch network is identical, the design complexity becomes simple.

Let us suppose that an interconnection network is composed of switch networks. To connect all the hosts, we must satisfy the following conditions:

1. Any host is connected to at most p switch networks;
2. Any two hosts are connected to switches in the same switch network.

Let v denote the number of switch networks. In general, the switch networks must have np edges in total, and hence the order of each switch network becomes np/v. Thus, we should increase the number of switch networks to reduce the order.

Let $X = \{x_0, x_1, \ldots, x_{v-1}\}$ be a set of switch networks. For each host, we can introduce a set of connected switch networks. This set can be the same for some hosts, so let us group hosts by the set of connected switch networks. We call this group a *host-group*. Regarding a host-group as a vertex, we can rewrite the conditions above as follows:

1. Any host-group is connected to exactly p distinct switch networks; and
2. Any two host-groups are connected to the same switch network.

We here assume any host-group is connected to *exactly* p switch networks because v should increase from the above.

4.4.2.1 Obvious Design

In the beginning, consider Euclidean plane. Let us regard a host-group as a line and a switch network as an intersection point of lines; then the number of intersection points of a line corresponds to the value of p. For the case of $p = 2$, we can draw Fig. 4.4a, which obviously satisfies the conditions above. As shown in Fig. 4.4a, this obvious design includes three host-groups and three switch networks when $p = 2$.

The obvious design with $p = 2$ can be extended with increasing p. We should add a new line (host-group) that joins no existing intersection points. Fig. 4.4b shows the result when we change p

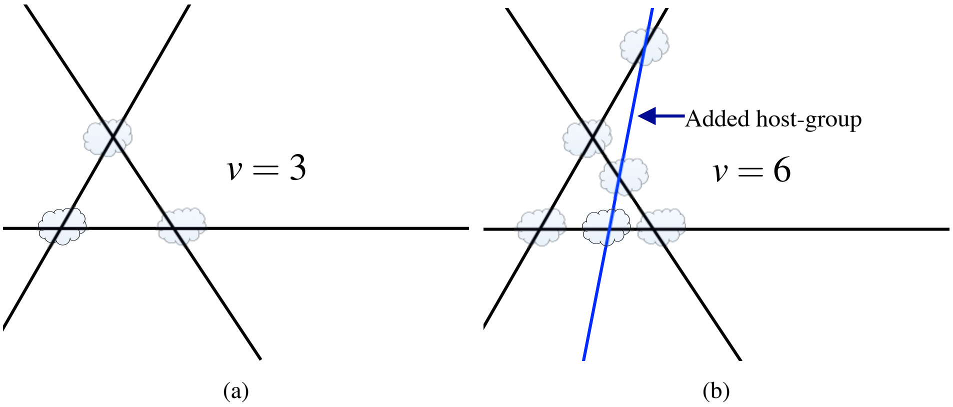


Figure 4.4: Obvious design represented by Euclidean plane. (a) $p = 2$. (b) $p = 3$.

from 2 to 3. As a result, v increases by p (i.e., 3 in Fig. 4.4b). In general, this obvious design consists of $p + 1$ host-groups and $1 + 2 + \cdots + p = p(p + 1)/2$ switch networks for $p \geqslant 2$ and satisfies the conditions above.

4.4.2.2 Proposed Design: Application of Finite Projective Planes

We propose a better design than an obvious one, the application of finite projective planes. A *finite projective plane* of *order* q consists of a set of $q^2 + q + 1$ points, a set of $q^2 + q + 1$ lines, and a relation between them, called *incidence*, such that:

1. Any two points are incident with exactly one line;

2. Any two lines determine exactly one point incident with both of them;

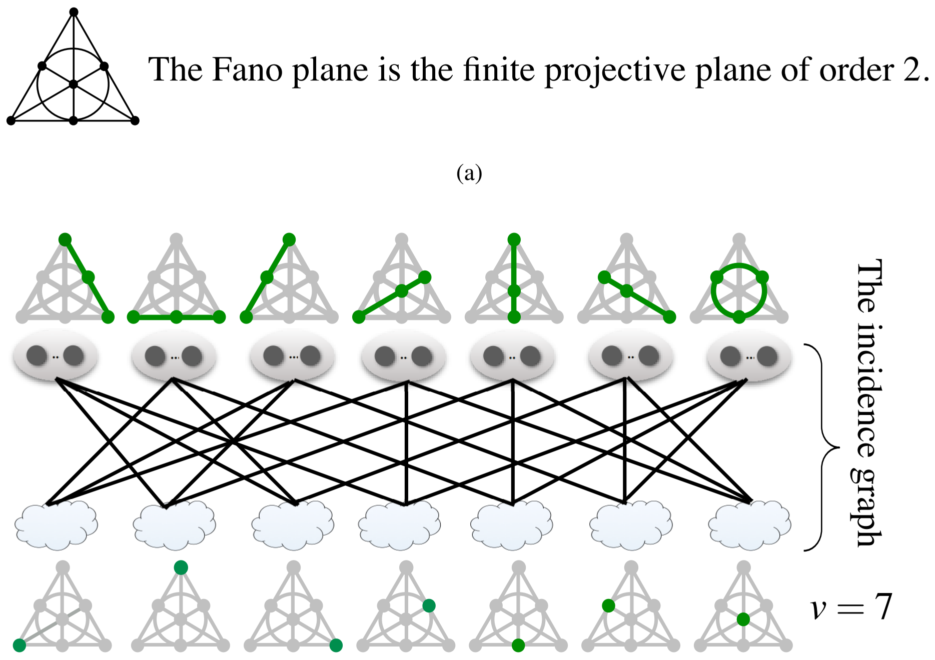
3. Every point has $q + 1$ lines on it; and

4. Every line contains $q + 1$ points.

From the definition, a finite projective plane of order $p - 1$ can be applied to a host-switch graph, regarding points and lines as switch-networks and host-groups, respectively.

For example, the finite projective plane of order 2, particularly called the Fano plane, consists of 7 points and 7 lines (Fig. 4.5a) [22]. For a finite projective plane, incidence can be represented by an incidence graph (a.k.a. a Levi graph [79]), as shown in Fig. 4.5b, and the incidence graph corresponds to a topology of host-groups and switch-networks. In general, by using a finite projective plane of order $p - 1$, we can design networks with $p^2 - p + 1$ host-groups and $p^2 - p + 1$ switch networks. Thus, the value of v is $(p^2 - 3p)/2 + 1$ greater than that of the obvious design above.

We can explain this design from different perspectives. Our solution, an incidence graph of a finite projective plane, also corresponds to a *balanced incomplete block design* (*BIBD*), studied in combinatorial mathematics [43]. Another typical BIBD is a finite affine plane, and we can also design networks using it. However, the value of v becomes greater than that when we use a finite projective



(a)

(b)

Figure 4.5: Proposed design applying a finite projective plane and its incidence graph ($p = 3$). (a) The Fano plane. (b) The incidence graph of the Fano plane, where a line and a point of the Fano plane correspond to a host-group and a switch network, respectively.

plane. Furthermore, an incidence graph of a finite projective plane is a solution of the degree/diameter problem for bipartite graphs of diameter 3 [111]: *given natural numbers Δ and D, find the largest possible number $B_{\Delta,D}$ of vertices in a bipartite graph of maximum degree d and diameter D.* We can regard a topology of host-groups and switch-networks as a bipartite graph. If the diameter is 3, any two host-groups can be connected to the same switch-network, and hence we can apply any bipartite graph of diameter 3 to networks with multi-port hosts. Note that a host-switch graph corresponds to permutated ND when the diameter of a bipartite graph is 2.

4.4.2.3 The Diameter and the h-ASPL

Clearly, we can state:

Lemma 4.1 (Upper bound on the diameter of a network with multi-port hosts)**.** *The diameter of a whole network is equal to or less than the diameter of a switch-network.*

This is why a network using small switch networks can reduce the diameter. Note that the h-ASPL of a whole network is possibly greater than the h-ASPL of a switch-network when all the shortest paths between hosts in the same host-group are small (some of them are unused although they are short paths). However, such cases are rare, and thus we can practically think the h-ASPL of a whole network is equal to or less than the h-ASPL of a switch-network.

Furthermore, we can optimize permutation for this network for reducing the h-ASPL. Let us again consider the time complexity as in Section 4.4.1. Since there exist $v = p^2 - p + 1$ switch networks, $|V|$ is equal to $m/(p^2 - p + 1)$. From the number of all the edges $(mr/2(p^2 - p + 1))$ and the number of host-links $(np/2(p^2 - p + 1))$, $|E|$ is equal to $(mr - np)/2(p^2 - p + 1)$. Thus, the time complexity in Step 1 is $\mathscr{O}((m^2r - mnp)/p^4)$. The number of possible shortest paths between two hosts is one between hosts in the different host-group and p between hosts in the same host-group. Thus, the time complexity in Step 2 is $\mathscr{O}(n^2p)$.

4.5 Evaluation

4.5.1 Experimental Setup

4.5.1.1 Topology

We construct host-switch graphs from the following switch topologies: the torus [37], the Fat-tree [12], the Dragonfly [71], and the randomly optimized topologies [124]. The former three topologies are current typical topologies used in supercomputers listed in TOP500 [3], while the last one is a state-of-the-art topology proposed for reducing the h-ASPL. We design five networks for each topology as follows: (1) LA; (2) ND; (3) permutated LA (denoted *p-LA*); (4) permutated ND (denoted *p-ND*); (5) application of finite projective planes (denoted *PP*). As a result, we compare 20 topologies in total (five designs for four switch topologies).

Unlike deterministic topologies that can be obtained immediately, heuristic topologies require an optimization algorithm and increase design complexity. We again use SA to decrease the h-ASPL as with [124]. To calculate the h-ASPL, we run the breadth-first search (BFS) from each switch. BFS takes $\mathscr{O}(|V||E|)$ running time for a graph (V, E), and it corresponds to $\mathscr{O}(m^2r - mn)$ in the case of a host-switch graph.

4.5.1.2 Routing

Our methods require routing in a host. In the case of p-LA and p-ND, the first hop of the shortest path is not obvious, and hence routing tables must include the routing information for all the destination hosts. In the case of PP, routing tables must include full routing information for only the destination hosts in the same group because the routing between the hosts in different groups depends on the host-group involving the destination host. Consequently, the routing tables become simple as compared with p-LA and p-ND. We discuss the size of routing tables in more detail in Section 4.5.5.

On the other hand, conventional two networks do not require routing tables in a host. LA never requires routing tables since the first hop is unique between a host and a switch. In the case of ND, the first hop affects the performance because of the congestion and load-balancing. However, ND requires no routing tables since the shortest path length is fixed regardless of the first hop.

For routing in a switch, we can use existing routing methods without any modification. In particular, we assume that we adopt a dead-lock free shortest path routing algorithm for arbitrary topologies. The previous literature [69, 112] shows that this routing is possible if we use enough virtual channels and full routing tables and it outperforms non-shortest path routing such as up*/down* routing [101]. We consider the size of routing tables in Section 4.5.

4.5.1.3 Floorplan

We assume a 1U switch and a rack-mounted host are located in a cabinet and connected via Mellanox InfiniBand FDR10 40Gbps QSFP cables. 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. Since our networks are irregular, we need to determine locations of the switches and the hosts.

The method for minimizing cable length when deploying a topology onto a floorplan of cabinets proceeds in two steps: clustering and mapping. In the first step, the vertices are clustered so that the number of edges between clusters becomes minimum. To cluster vertices in a cabinet is equivalent to converting the graph into a weighted graph by merging several vertices where loop edges are removed, and multiple edges are a single weighted edge. We develop hierarchical clustering methods, modifying them so that the resulting cluster size does not exceed the specified cabinet size. We adopt the Ward method [119], which produces excellent results according to [57].

In the second step, the physical layout is determined by mapping each cluster to a cabinet on a floorplan so that the total cable length becomes minimum. This problem corresponds to the quadratic assignment problem (QAP) [52]. Since QAP is NP-complete [99], we adopt the robust taboo search implemented for QAP by Taillard [117], which produces almost the same results as those by SA [57].

4.5.2 Design Complexity

The design complexity depends on the time complexity of calculating h-ASPL, which we have shown in Section 4.4. Table 4.1 summarizes the time complexity. In Step 1, LA and p-LA require the largest time complexity since they include a massive switch network. This complexity is negligible in the case of conventional topologies. In the case of randomly optimized topologies, however, it is critical because it determines the running time of SA. On the other hand, PP requires drastically little complexity, and consequently PP is suitable for randomly optimized topologies. The time complexity in Step 2 is needed only for our methods. Obviously, p-LA requires the longest time for optimized permutation; $\mathcal{O}(p)$ time longer time than time for p-ND and PP. Overall, p-LA requires the worst design complexity, and PP requires the least design complexity.

Table 4.1: Time complexity for Step 1 (calculating ASPL between switches in a switch network) and Step 2 (calculating h-ASPL between hosts on the basis of the results of Step 1).

	Step 1	**Step 2**
LA	$\mathcal{O}(m^2r - mnp)$	–
ND	$\mathcal{O}\left(\frac{m^2r-mnp}{p^2}\right)$	–
p-LA	$\mathcal{O}(m^2r - mnp)$	$\mathcal{O}(n^2p^2)$
p-ND	$\mathcal{O}\left(\frac{m^2r-mnp}{p^2}\right)$	$\mathcal{O}(n^2p)$
PP	$\mathcal{O}\left(\frac{m^2r-mnp}{p^4}\right)$	$\mathcal{O}(n^2p)$

4.5.3 Topological Properties

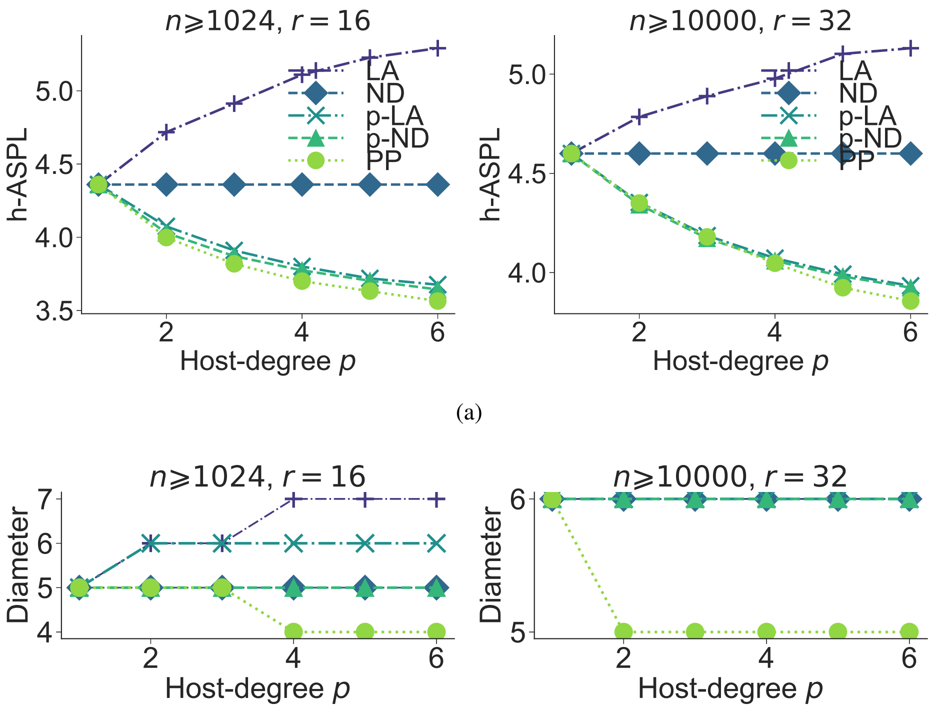
4.5.3.1 Hop Count: Host-to-Host Average Shortest Path Length (h-ASPL) and Diameter

Fig. 4.6a shows the relationship between the h-ASPL and the host-degree p of randomly optimized topologies. The h-ASPL for ND is naturally fixed. For LA, the h-ASPL increases logarithmically as p increases since switch networks get larger and more host-switch links are needed. For the remainder, the h-ASPL decreases exponentially because of the optimized permutation and the smallness of switch networks. In particular, the difference between LA and p-LA shows the impact of the optimized permutation. The results indicate that PP reduces the h-ASPL the most, followed in order by p-ND and p-LA. When $n \geqslant 10000$ and $r = 32$, the results are slightly different because the Moore bound (the lower bound of the h-ASPL [124]) depends on n and r, but the tendency is similar.

The diameter, shown in Fig. 4.6b, changes differently from the h-ASPL, except for the cases of ND. For LA and p-LA, the diameter becomes greater with increasing p since switch networks get larger. It is notable that, for p-LA, the h-ASPL and the diameter grow conversely. p-ND holds the diameter constant while the h-ASPL decreases. When $n \geqslant 10000$ and $r = 32$, there exist no cases that the diameter increases because the Moore bound on the diameter seldom increase when n and r are sufficiently large. However, only PP decreases the diameter with increasing p since switch networks get smaller.

Fig. 4.7a shows the relationship between the h-ASPL and the host-degree p of torus topologies. Results for ND and p-ND are similar to those of randomly optimized topologies (Fig. 4.6a). For LA, p-LA, and PP, however, results are distinctly different from those of randomly optimized topologies; the h-ASPL has no clear relationship with the value of p. In some cases—e.g., $p = 4$ for PP when $n \geqslant 1024$—plots seem to be outliers. This is because there exist no torus topologies with the required order in some cases and we use torus topologies which are big enough to connect the required number of hosts. Except for such cases, p-LA, p-ND, and PP provide the similar h-ASPL.

Results of the diameter, shown in Fig. 4.7b, are entirely different from those of randomly optimized topologies (Fig. 4.6b). When $n \geqslant 1024$ and $r = 16$, p-LA and p-ND decrease the diameter with increasing p as well as PP. This indicates the effectiveness of the optimized permutation for structured



(a)

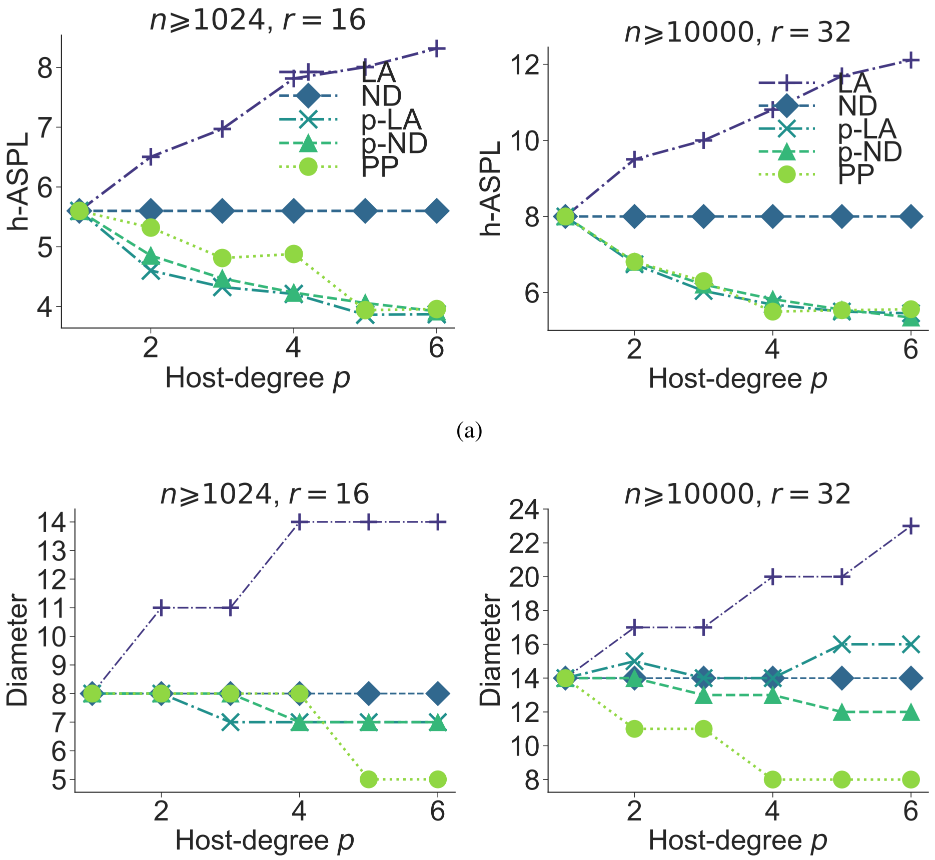
(b)

Figure 4.6: Hop count of randomly optimized topologies with p-port hosts $(0 \leqslant p \leqslant 6)$ when $(n, r) = (1024, 16)$ and $(n, r) = (10000, 32)$: (a) h-ASPL and (b) diameter.

topologies. For PP, the diameter does not decrease when p is up to four, and then it decreases drastically. This drastic change is caused by the characteristics of torus topologies; note that the diameter of the ring structure in torus topologies changes as the number of nodes in the ring structure increases by two, not one. When $n \geqslant 10000$ and $r = 32$, p-LA increases the diameter. We consider this is because the number of hosts per switch (n/m) increases; in this evaluation, we increase the radix as well as the number of hosts, but do not increase the dimension of torus topologies. In contrast, p-ND and PP decrease the diameter more drastically as compared with when $n \geqslant 1024$ and $r = 16$.

Thus far, we evaluate the topologies that allow us to set the radix r arbitrarily, but the following topologies require a specific radix according to the number of connected hosts. As a result, we can observe different scaling.

Fig. 4.8a shows the relationship between the h-ASPL and the host-degree p of Dragonfly topologies. Since the radix increases as well as the size of a switch network increases, the h-ASPL for LA is almost fixed. The results show that p-LA reduces the h-ASPL with increasing p the most, followed in order by p-ND and PP; this order is opposite to the order in the case of randomly optimized topologies (Fig. 4.6a). Furthermore, the differences between them are more significant. These are obviously because the radix r of Dragonfly topologies changes according to the number of connected hosts. As shown in Fig. 4.8c, PP reduces the radix with increasing p while p-LA increases it since PP and p-LA



(a)

(b)

Figure 4.7: Hop count of torus topologies with p-port hosts ($0 \leqslant p \leqslant 6$) when $n \geqslant 1024$ and $n \geqslant 10000$: (a) h-ASPL and (b) diameter.

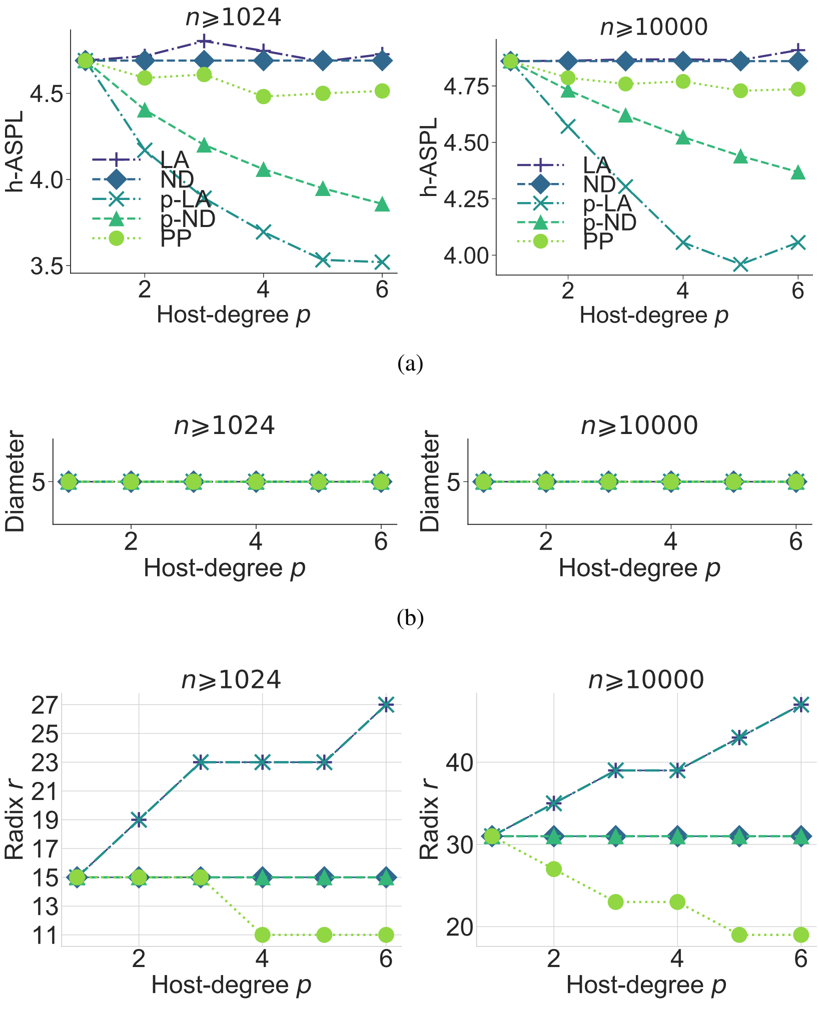
change the size of switch networks. It would be notable that PP can reduce both the h-ASPL and the radix.

The diameter, shown in Fig. 4.8b, is constant in all the cases—even if the order n changes from 1024 to 10000. The diameter of Dragonfly topologies is always three if the groups of routers in a Dragonfly constitute a clique (a complete graph) [71], so the diameter including host-switch hops becomes five.

Fig. 4.9a shows the relationship between the h-ASPL and the host-degree p of Fat-tree topologies. The results are similar to those of Dragonfly topologies since Fat-tree topologies also have the fixed diameter and the variable radix. Also, the diameter, shown in Fig. 4.9b, is constant in all the cases as with Dragonfly topologies.

4.5.3.2 Bisection Width (BiW)

Fig. 4.10 shows the relationship between bisection widths (BiWs) and the host-degree p of the four topologies. Obviously, BiW becomes larger with increasing p since the number of switches and



(a)

(b)

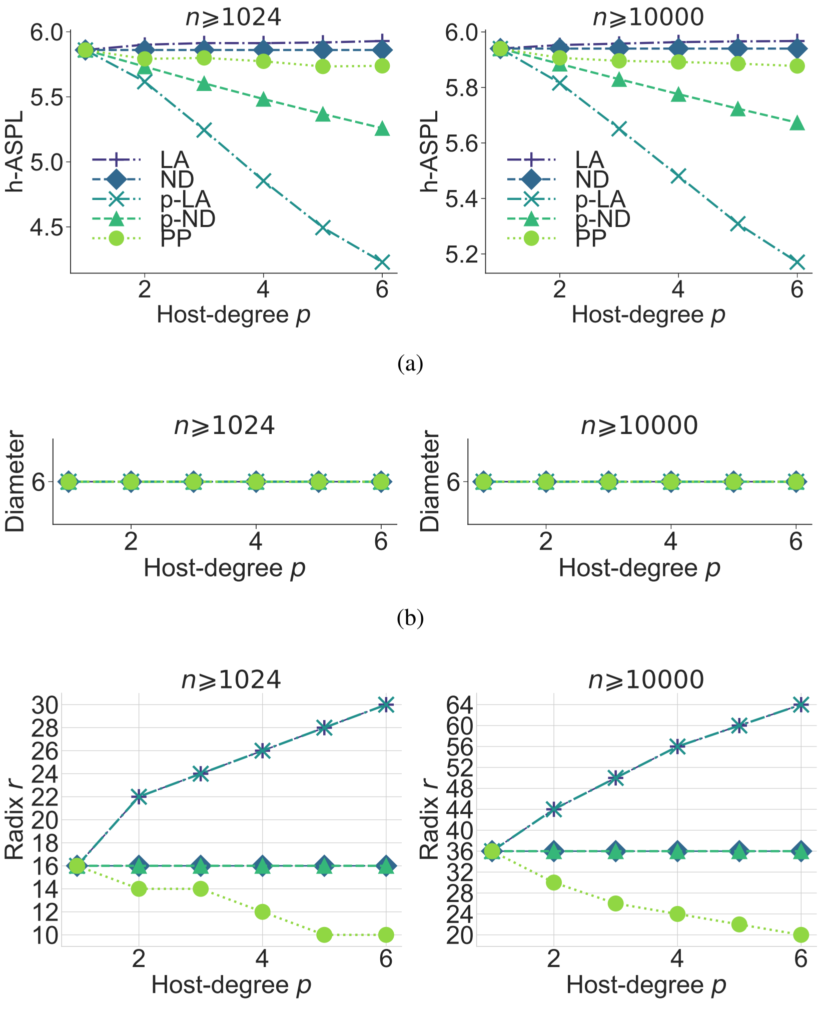
(c)

Figure 4.8: Hop count of Dragonfly topologies with p-port hosts ($0 \leqslant p \leqslant 6$) when $n \geqslant 1024$ and $n \geqslant 10000$: (a) h-ASPL; (b) diameter; and (c) radix. Note that the Dragonfly changes radix according to the required number of connected hosts.

links increases. The increases are almost linear, and their gradients depend on the topology and the network design. Let us consider the results of each topology below.

The results of randomly optimized topologies are shown in Fig. 4.10a. When $n \geqslant 10000$ and $r = 32$, p-LA and p-ND provide high BiW while LA, ND, and PP provide low BiW. This suggests that the optimized permutation for large switch networks increases BiW. When $n \geqslant 1024$ and $r = 16$, the results are similar to the above, but they differ from the above in that p-LA provides higher BiW than p-ND does. We consider this is because p-LA increases the diameter.

The results of torus topologies are shown in Fig. 4.10b. Again, p-LA and p-ND provide high BiW while LA, ND, and PP provide low BiW. It is notable that the BiWs for LA and ND seem significantly small; even when p is six, the BiWs fall below the full BiW ($n/2$). This is because the BiWs of torus topologies are originally small and ND provides a linear increase (i.e., it provides p times larger BiW



(a)

(b)

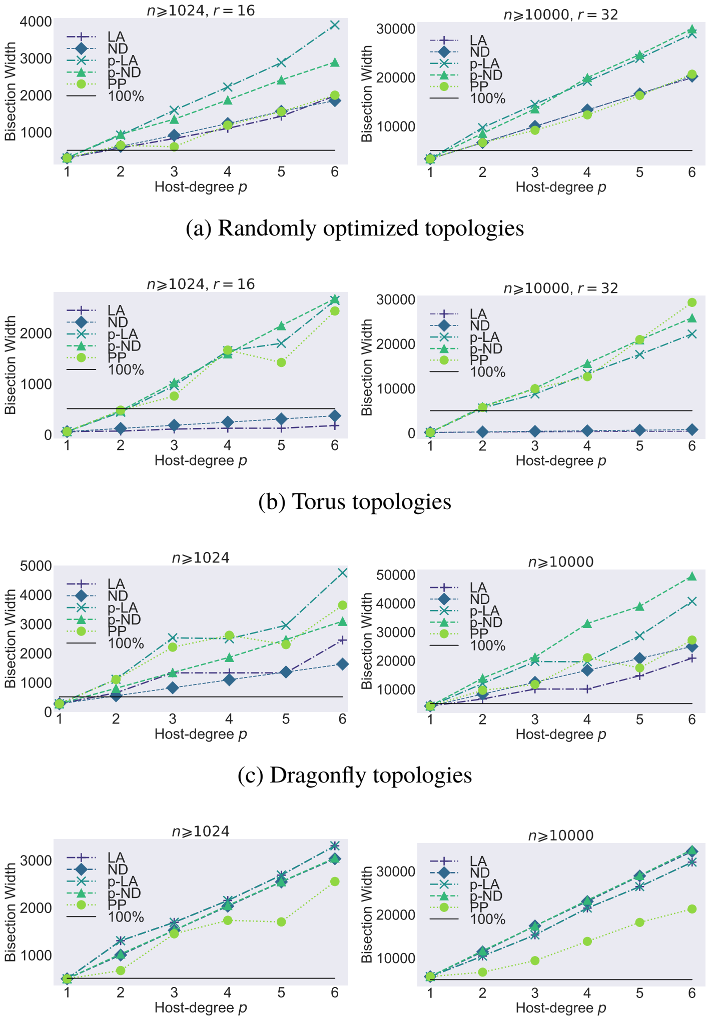
(c)

Figure 4.9: Hop count of Fat-tree topologies with p-port hosts ($0 \leqslant p \leqslant 6$) when $n \geqslant 1024$ and $n \geqslant 10000$: (a) h-ASPL; (b) diameter; and (c) radix. Note that the fat-tree changes radix according to the required number of connected hosts.

as compared with an original torus topology with $p = 1$). The BiW for LA grows more slowly than the linear increase. Thus, our results indicate that the optimized permutation drastically increases the BiW when the BiW of an original topology is small. For p-LA and PP, some plots seem to be outliers since there exist no Dragonfly topologies with required order and we use the topologies which are big enough to connect the required number of hosts.

The results of Dragonfly topologies, shown in Fig. 4.10c, show different tendencies from results above. ND provides the linear increase, and the optimized permutation increases the BiW, but there exists no clear relationship between the BiW and p for the rest. This is because the Dragonfly changes the radix according to the required number connected hosts and consequently also the BiW changes.

The results of Fat-tree topologies are shown in Fig. 4.10d. Note that Fat-tree topologies we adopt are full-bisection and their BiW is large even when $p = 1$. For that reason, in contrast with the torus



(a) Randomly optimized topologies

(b) Torus topologies

(c) Dragonfly topologies

(d) Fat-tree topologies

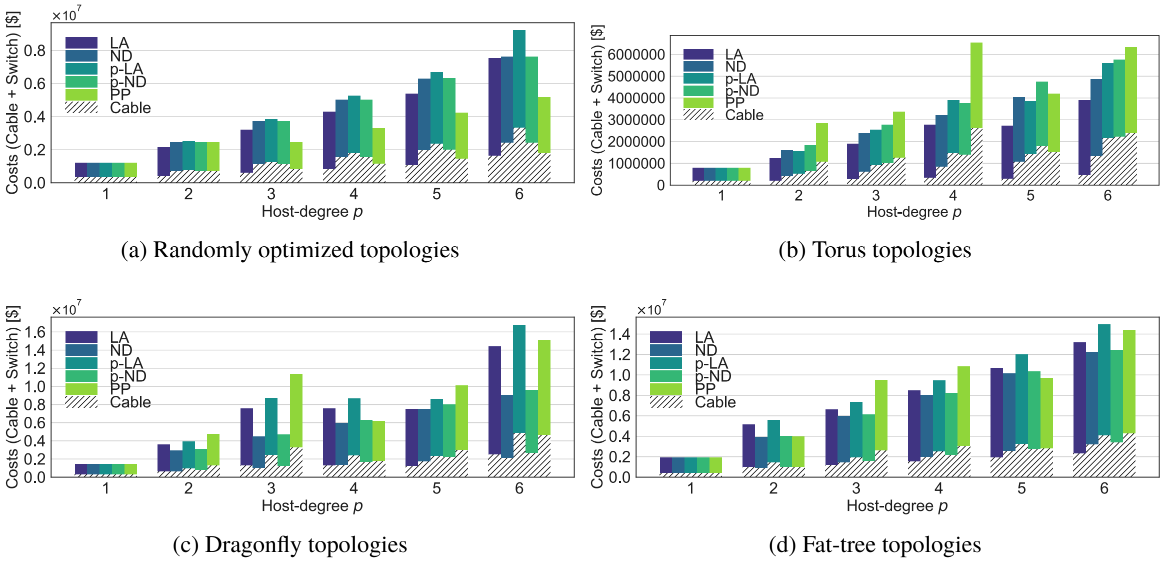
Figure 4.10: Bisection widths. The line denoted by “100%” shows full bisection width, i.e., $n/2$.

topologies, the optimized permutation cannot increase the BiW as compared with the BiW for ND; note that the results for ND and p-ND are almost the same, also for LA and p-LA. In the case of PP, the BiW is below the BiW for ND because the number of switches for PP is small, which is preferable in terms of costs.

Overall, our results demonstrate the following: (1) ND provides the linear increase of the BiW; (2) the optimized permutation provides higher BiW than ND does when an original BiW when $p = 1$ is small; and (3) PP provides slightly lower BiW than p-LA and p-ND do.

4.5.4 Costs

Costs are estimated by models of Mellanox InfiniBand FDR10 switches and Mellanox InfiniBand FDR10 40Gbps QSFP cables [21]. Fig. 4.11 shows the relationship between costs and the host-degree p of the four topologies when $n \geqslant 1024$. Obviously costs become larger with increasing p as well



(a) Randomly optimized topologies

(b) Torus topologies

(c) Dragonfly topologies

(d) Fat-tree topologies

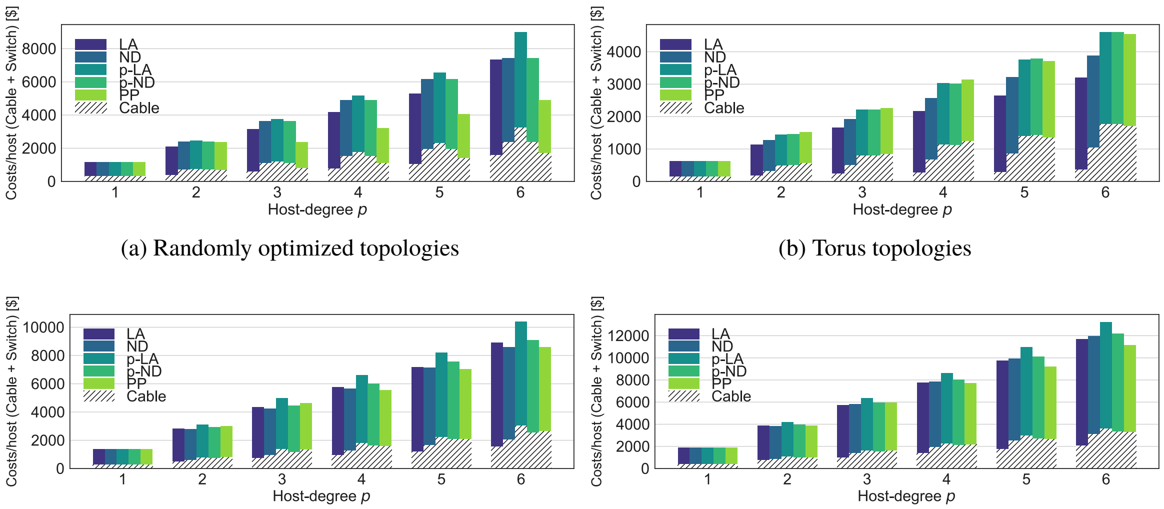
Figure 4.11: Costs of cable and switches when $n \geqslant 1024$: (a) Randomly optimized topologies; (b) Torus topologies; (c) Dragonfly topologies; and (d) Fat-tree topologies. Note that the number of hosts is not fixed; costs per host is shown in Fig. 4.12.

as the BiW. The increasing tendency for each case is almost consistent with that in Fig. 4.10 (for example, sudden decrease for PP of Dragonfly topologies when $p = 4$). However, the results in Fig. 4.11 do not provide information about cost efficiency since the number of connected hosts is not completely fixed. Thus, we subsequently evaluate costs per host.

Costs per host are shown in Fig. 4.12. The results of randomly-optimized, Dragonfly, and Fat-tree topologies are similar to each other; p-LA provides the highest cost. The differences exist mainly in cable costs. The optimized permutation requires long cables, and hence p-LA and p-ND need high costs. On the other hand, LA, ND, and PP do not require long cables. In particular, the cable costs of LA are small since LA requires no long host-switch links. Interestingly, PP requires lower costs than LA and ND do. This suggests that PP is highly cost-effective methods for reducing the hop count. The results of torus topologies, shown in Fig. 4.12b, provide different tendencies; p-LA, p-ND, and PP require similar costs. We consider this is because cables in a single switch network are short while cables that connect nodes in different switch networks may become long. That is why PP does not save costs in the case of torus topologies.

4.5.5 Size of Routing Tables

Randomly optimized topologies require routing tables for using the shortest path lengths, even when $p = 1$. Let us consider the cases of LA and ND. For a host-switch graph with single-port hosts, each switch must have information about the output link for n destination hosts, each of which costs $\mathscr{O}(\log r)$. Hence, a host-switch graph requires $\mathscr{O}(mn \log r)$ information in a routing table in total.



(a) Randomly optimized topologies

(b) Torus topologies

(c) Dragonfly topologies

(d) Fat-tree topologies

Figure 4.12: Costs per host of cable and switches when $n \geqslant 1024$: (a) Randomly optimized topologies; (b) Torus topologies; (c) Dragonfly topologies; and (d) Fat-tree topologies.

Table 4.2: Total size of routing tables.

	In switches	**In hosts**
LA	$\mathscr{O}(mn \log r)$	–
ND	$\mathscr{O}(mn \log r)$	–
p-LA	$\mathscr{O}(mn \log r)$	$\mathscr{O}(n^2 \log p)$
p-ND	$\mathscr{O}(mn \log r)$	$\mathscr{O}(n^2 \log p)$
PP	$\mathscr{O}\left(\frac{mn \log r}{p}\right)$	$\mathscr{O}\left(\frac{(n^2+p^4)\log p}{p^2}\right)$

However, PP reduces the size of routing tables since each switch has only np/v possible destination hosts. As a result, PP requires $\mathscr{O}((mn \log r)/p)$ information in total (note that v is equal to $p^2 - p + 1$).

Additionally, p-LA, p-ND, and PP require routing tables regardless of the switch topology for a host since there exist p possible first hops. Each host must have information about the output link for n destination hosts, each of which costs $\mathscr{O}(\log p)$. Thus, the total size of routing tables in the hosts becomes $\mathscr{O}(n^2 \log p)$. However, in the case of PP, the communication between hosts in different host groups only requires information about an incidence graph of a finite projective plane (described in Section 4.4), and hence the size of routing tables is saved as compared with p-LA and p-ND. More specifically, each host for PP requires $\mathscr{O}(n^2 \log p)$ information for n/v hosts in the same group and $\mathscr{O}(v \log p)$ information for $n(v-1)/v$ hosts in the different groups. Thus, the total size of routing tables in the hosts becomes $\mathscr{O}(((n^2 + p^4) \log p)/p^2)$.

The results above are summarized in Table 4.2. Naturally, LA and ND require no overheads. In contrast, the optimized permutation requires the largest overheads. PP reduces the overheads as well as the design complexity.

Table 4.3: Improvement rate of simulated MPI performance relative to the case when $p = 1$ (%).

Topology	p	LA	ND	p-LA	p-ND	PP
Randomly optimized	2	-5.7	1.7	13	12	14
	6	-17	2.5	45	45	54
Torus	2	-7.4	8.1	22	14	-5.0
	6	-22	13	74	74	51
Dragonfly	2	-23	5.4	-22	-25	-25
	6	-21	9.3	4.0	0.39	-18
Fat-tree	2	-1.7	25	103	73	61
	6	-2.7	64	244	187	162

4.5.6 Discrete-Event Simulation

Finally, we evaluate the MPI performance when using each network by SimGrid discrete-event simulator [32]. 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 switch has a 60 nsec delay, each link has a 36 nsec delay and a 10 Gbps bandwidth, and each host has a computation speed of 100 GFLOPS. We configure SimGrid to use its built-in version of the MVAPICH2 implementation of MPI collective communications. We use four kernels (the conjugate gradient, the multi-grid on a sequence of meshes, the discrete 3D fast Fourier transform, and embarrassingly parallel) and three pseudo-applications (the block tridiagonal solver, the scalar pentadiagonal solver, and the lower-upper Gauss-Seidel solver). The codes are from NAS parallel benchmarks [8] (v3.3.1, Class A for kernels and Class B for pseudo-applications).

Table 4.3 shows results of $p = 2, 6$ when $n \geqslant 1024$, the geometric mean of the improvement rate relative to the case when $p = 1$. Obviously, LA decreases the performance in all the cases because it increases the diameter and the h-ASPL. This indicates that LA is not suitable for high-performance interconnection networks.

For randomly optimized topologies, ND and all of our methods improve the performance. The improvement rate of ND shows the effect of BiW improvement, and the hop count reduction further enhances the performance. From the results, we can say that the less hop count, the more performance for randomly optimized topologies.

For torus topologies, PP for $p = 2$ decreases the performance because it reduces the hop count only slightly and destroys regularity of the torus. When $p = 6$, PP increases the performance since the reduction of the hop count becomes significant. However, p-LA and p-ND provide more improvements. This suggests that the optimized permutation drastically affects structured topologies, rather than random topologies.

Somewhat surprisingly, our methods can hardly improve Dragonfly topologies. This is because Dragonfly topologies are near optimal [124] in terms of the h-ASPL and besides they are regularly structured. In particular, p-LA and p-ND increase the radix without improving the performance, so networks with $p = 1$ are better than networks with multi-port hosts. On the other hand, PP can be used to decrease the radix at the sacrifice of the performance. Only ND significantly improves the performance since it increases the maximum throughput by p times.

For Fat-tree topologies, our proposed methods can improve the performance drastically—up to 244%. This is because the hop count drastically decreases, particularly in the case of p-LA. This suggests that using multi-port hosts is suitable for topologies with the large diameter, although also costs drastically increase. In particular, it would be notable that PP decreases the radix while it improves the performance.

From the results above, we should carefully select the network design according to the switch topology. In conclusion, we suggest the following selection for each switch topology.

- For randomly optimized topologies, PP is a good selection in terms of both costs and MPI performance.
- For torus topologies, p-LA or p-ND are good selections in terms of the MPI performance.
- For Dragonfly topologies, ND is a good selection in terms of the MPI performance.
- For Fat-tree topologies, p-LA is apparently a good selection in terms of the MPI performance if we do not consider costs and the radix. However, we suggest that PP is a better selection because it decreases the radix while it also improves the performance, and it requires fewer costs than p-LA does.

4.6 Related Work

4.6.1 Direct Networks

TOP500 includes supercomputers such as Sequoia (BlueGene/Q) [88] and Titan [25], which integrate both routers and processor cores (and also memories) into a compute node. Anton [44] supports a 3-D torus off-chip network by integrating six on-chip routers that connect off-chip links to an ASIC-chip node. In the DCN domain, the DCell architecture [61] and the CamCube [10] architecture forward packets in servers in a fully connected interconnect and a 3-D torus interconnect, respectively. Such networks are called *direct* networks, i.e., a switch and a certain number of hosts are regarded as a node that constitutes a network. We are able to regard such networks as a switch network, and our methods can extend them by using multi-port hosts to reduce the end-to-end latency.

4.6.2 Packet Forwarding in Hosts

In this chapter, we assume the hosts never forward packets as described in Section 4.2.2. However, several systems use multi-port hosts for forwarding packets, having hosts act as intermediate switches.

The PACS-CS supercomputer uses a multi-dimensional hyper-crossbar that routes packets at each host equipped with multiple network interface cards [115]. In the DCN domain, the BCube architecture forwards packets in both switches and servers [60]. Although such networks have both hosts and switches, there exist no host-switch graphs identical to such networks. Furthermore, they would require costs overheads for packet forwarding in hosts. Thus, we cannot fairly compare such networks and the networks studied in this chapter.

4.6.3 On-Chip Networks with Multi-Port Hosts

Several researchers propose using multi-port hosts for low-latency on-chip networks. Matsutani *et al.* propose Fat H-Tree [82] as an attractive alternative to tree-based networks such as the fat-tree in a microarchitecture domain. It uses multi-port hosts to combine two folded H-tree networks and consequently it provides a torus structure and reduces hop count. The authors report that using multi-port hosts reduces energy consumption by improving the performance while it requires overhead costs. Fat H-Tree also forwards packets in a host, and thus it is out of a host-switch graph.

Kawano *et al.* propose optimized core-links for low-latency on-chip networks [68]. They add links, called core-links, between a host and a switch to the mesh topology. The topology of core-links is optimized by a genetic algorithm (GA). The authors report that adding core-links reduce the average latency for the synthetic bit complement traffic by 48%. This proposal uses one switch network and a kind of twisting (they use GA instead of SA) and thus belongs to p-LA. Note that they *add* links between a host and a switch, i.e., increase the radix of a switch, and thus it is self-evident that the hop count decreases by using multi-port hosts. On the other hand, this chapter demonstrates that using multi-port hosts reduces hop count without increasing radix; on the contrary, SND reduces hop count while it decreases the radix in some cases (see Fig. 4.7–4.9).

Camacho and Flich propose HPC-Mesh [30], the homogeneous parallel concentrated mesh topology with an intelligent injection algorithm. It provides four disjoint homogeneous concentrated mesh networks and each host is connected with all the four networks. Thus it belongs to ND. The authors report that multi-port hosts improve the performance and the fault tolerance.

In on-chip networks, multi-port hosts are also used for improving energy efficiency. Catnap [40] uses multiple networks-on-chip, denoted by `Multi-NoC`, with power gating for reducing power consumption. In `Multi-NoC`, a single network interface is connected to several routers, each of which belongs to a different network (called a *subnet*). This system thus belongs to ND, and hence it never reduces hop count.

4.6.4 Low-Latency Networks with Single-Port Hosts

Most of the previous research for low-latency interconnection networks focuses on reducing the diameter or the average shortest path length (ASPL) when single-port hosts are used. To this end, an interconnection network is represented as an undirected graph, and some graph-theoretic problems are solved. The degree/diameter problem is a classical problem of finding the largest number of vertices in a graph of given maximum degree and diameter. Its solutions—such as the MMS graph and the polarity graph (a.k.a. Brown's construction or Brown graph)—are candidates for topologies of interconnection networks [21,87]. Another problem is the order/degree problem, which is a relatively new problem of finding the smallest diameter in a graph of given order and degree [5]. The previous research [124] shows that randomized algorithm such as simulated annealing is efficient for solving this problem.

Projective networks [31] are alternative methods for designing low-latency interconnection networks different from the methods above. Rather interestingly, they use incidence graphs of finite projective planes for interconnection networks with single-port hosts, in contrast with our application for interconnection networks with multi-port hosts. Furthermore, their focus is not only on the hop count but also the network utilization and costs.

Our proposed methods can directly adopt the topologies above as switch topologies and extend them by using multi-port hosts.

4.7 Summary

This chapter has studied interconnection networks with multi-port hosts in the context of the impacts of adding ports to hosts on end-to-end latencies. We model them as a *host-switch graph* with host-degree p and study five designs using conventional and novel methods. Conventional two methods include link aggregation (LA) and network duplication (ND), which are used for improving bandwidths and throughput rather than end-to-end latency. We have proposed two novel methods: the permutation of host-switch mapping and the application of finite projective planes. We apply the permutation for LA and ND (denoted by *p-LA* and *p-ND*, respectively) and demonstrate that the permutation efficiently reduces the hop count. The networks applying a finite projective plane (denoted by *PP*) reduce the size of switch networks, and consequently the hop count decreases.

We have evaluated 20 networks (five designs for four switch topologies) in terms of the design complexity, the host-to-host average shortest path length (h-ASPL), the diameter, the bisection width (BiW), the size of routing tables, and costs. Our experimental results show that using multi-port hosts increases the BiW and costs almost linearly. The optimized permutation (p-LA and p-ND) and applying a finite projective plane (PP) exponentially reduce the h-ASPL for both random and non-random topologies. In particular, we conclude that PP is a cost-effective method in terms of the design complexity, costs, and the size of routing tables, especially for randomly-optimized and Fat-tree topologies.

Chapter 5

Conclusions

5.1 Summary

In this dissertation, we have studied low end-to-end network topologies with low hop count. In Chapter 2, we have surveyed interconnection networks, graph theory, network science, and design theory. All of these independent studies are used for designing high-performance interconnection networks. In Chapter 3, we have introduced a host-switch graph and the order/degree problem. We have established a design method for reducing the host-to-host average shortest path length (h-ASPL) and increasing the bisection width (BiW). We have obtained interesting findings, including the optimal number of switches and the relationship between the h-ASPL and the BiW. In Chapter 4, we have extended a host-switch graph so that it represents a network with multi-port hosts. We have then proposed two methods for reducing the h-ASPL of an interconnection networks with multi-port hosts: the permutation of host-switch mapping and the application of the finite projective plane (one of the balanced incomplete block designs). The proposed methods have been evaluated theoretically and experimentally.

5.2 Future Directions

5.2.1 Theoretical and Practical Extensions

This dissertation would motivate further research from both theoretical and practical aspects. Possible future work includes

- proving the optimality of the number of switches such that the continuous Moore bound becomes minimum,
- designing more efficient algorithms for optimizing the h-ASPL and the BiW,
- deriving the optimal host distribution, and
- deriving the optimal host-switch mapping for networks with multi-port hosts.

5.2.2 Routing methods including Virtual 1-D Mapping

Also, future work should focus on routing methods. In Chapter 2, we classified studies of interconnection networks into three categories: topology, routing, and layout. In this context, the focus of this dissertation was on topology and layout; we simply assume a deterministic shortest-path routing. We now briefly describe possible study on routing based on a host-switch graph.

5.2.3 Topology-Routing Co-Optimization

Furthermore, we suggest possibility of a novel design methodology, *topology-routing co-optimization* (*TRCO*). This concept is similar to the concept of randomly optimized grid graph (see Section 2.1.3.2), which is so to speak a topology-layout co-optimization. Moreover, a 1-D mapping can possibly include information about the layout. We might ultimately reach a *topology-routing-layout co-optimization*[1].

5.3 Concluding Remarks

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. This dissertation substantiates 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. Also, when a host has multiple ports, this dissertation shows that projective planes enable us to increase the performance of an interconnection network. Thus, the study of host-switch graphs bridges a gap between graph theory and computer engineering.

[1] Note that, intuitively, a 1-D mapping such that link-length is small provides a good deadlock-free routing because the number of turns (a change of forwarding directions at a node) becomes small.

Appendix A

Theorems

A.1 Optimality of a Clique Host-Switch Graph

Let a *clique host-switch graph* denote a host-switch graph such that all of the switches constitute a clique. For any clique host-switch graph, each switch must be connected with exactly $m - 1$ switches. Here, we prove that a host-switch graph with the lowest h-ASPL for given n and r is a clique host-switch graph.

First, we have the following lemma.

Lemma A.1. *A clique host-switch graph with the lowest h-ASPL has the minimum possible number of switches.*

Proof. Suppose that a clique host-switch graph with the lowest h-ASPL G has m switches while a clique host-switch graph with m' switches ($m' < m$) can be constructed. We can then reduce h-ASPL of G by removing a switch s_i and reconnect hosts connected with s_i to another switch, a contradiction. □

Lemma A.1 leads to:

Corollary A.1. *Let k_i denote the number of hosts connected with s_i. A clique host-switch graph with the lowest h-ASPL satisfies $k_i \geqslant m$ for all i ($0 \leqslant i \leqslant m - 1$).*

Here, we can derive the following theorem:

Theorem A.1. *For fixed n and r, there exists a clique host-switch graph that has the lowest h-ASPL.*

Proof. Let G be a clique host-switch graph such that $k_i \geqslant m$ for all i ($0 \leqslant i \leqslant m-1$) and the number of switches is minimum. From Lemma A.1, G is a clique host-switch graph with the lowest h-ASPL. Let G' be a host-switch graph with parameters n, m', and r, which is not a clique host-switch graph. Let us consider construct G' from G, and compare $A(G)$ and $A(G')$:

Case 1: $m' > m$

Trivially, $A(G') \geqslant A(G)$ since we cannot increase k_i.

Case 2: $m' \leqslant m$

To increase k_i, we must cut an edge (s_i, s_j) $(i \neq j)$ and reconnect a host to s_i. If we cut the edge, then the h-ASPL increases by at least $k_i \cdot k_j / \binom{n}{2}$. If we reconnect a host to s_i, then the h-ASPL decreases by at most $k_i / \binom{n}{2}$. Hence, the h-ASPL does not increase only if $k_j < 2$. From Corollary A.1, G satisfies $k_j > m$, and G' satisfies $k_j < 2$ only after cutting at least $m - 2$ edges connected with s_i. After cutting $m - 2$ edges, however, s_i has only one edge, and hence we cannot cut an edge any more. Therefore, $A(G') \geqslant A(G)$ holds. □

Bibliography

[1] InfiniBand architecture specification volume 2 release 1.3. https://www.infinibandta.org/ibta-specification/, 2012.

[2] Data intensive computing, the 3rd wall, and the need for innovation in architecture. http://extremecomputingtraining.anl.gov/files/2017/08/ATPESC_2017_Dinner_Talk_6_8-4_Kogge-Data_Intensive_Computing.pdf, 2017.

[3] November 2017 | TOP500 Supercomputer Sites. https://www.top500.org/lists/2017/11/, 2017.

[4] Graph 500 | large-scale benchmarks. https://graph500.org/, 2018.

[5] GraphGolf: the order/degree problem competition. http://research.nii.ac.jp/graphgolf/, 2018.

[6] InfiniBand EDR 100Gb/s Switch System. http://www.mellanox.com/related-docs/prod_ib_switch_systems/pb_sb7800.pdf, 2018.

[7] International Roadmap for Devices and Systems (IRDS) 2017 Edition. https://irds.ieee.org/roadmap-2017, 2018.

[8] The NASA Advanced Supercomputing (NAS) Parallel Benchmarks. http://www.nas.nasa.gov/Software/NPB/, 2018.

[9] Dennis Abts and Bob Felderman. A guided tour of data-center networking. *ACM Commun.*, 55(6):44–51, 2012.

[10] Hussam Abu-Libdeh, Paolo Costa, Antony Rowstron, Greg O'Shea, and Austin Donnelly. Symbiotic routing in future data centers. *ACM SIGCOMM Conf. Data Commun.*, pages 51–62, Aug. 2010.

[11] Narasimha R Adiga et al. Blue Gene/L torus interconnection network. *IBM J. Research and Development*, 49:265–276, 2005.

[12] Mohammad Al-Fares, Alexander Loukissas, and Amin Vahdat. A scalable, commodity data center network architecture. In *Proc. ACM SIGCOMM Conf.*, pages 63–74, Aug. 2008.

[13] Sadaf R Alam, Themis Athanassiadou, Timothy W Robinson, Gilles Fourestey, Andreas Jocksch, Luca Marsella, Jean-Guillaume Piccinali, Jeff Poznanovic, Benjamin Cumming, and Dominik Ulmer. First 12-cabinets Cray XC30 system at CSCS: Scaling and performance efficiencies of applications. In *Proc. Cray User Group Conf.*, May 2013.

[14] Réka Albert, Hawoong Jeong, and Albert-László Barabási. Diameter of the World-Wide Web. *Nature*, 401(6749):130–131, 1999.

[15] Katie Antypas, Nicholas Wright, Nicholas Cardo, Allison Andrews, and Matthew Cordery. Cori: a Cray XC pre-exascale system for NERSC. In *Proc. Cray User Group Conf.*, May 2014.

[16] Jordi Arjona Aroca and Antonio Fernández Anta. Bisection (band)width of product networks with application to data centers. *IEEE Trans. Parallel Distrib. Syst.*, 25:570–580, Mar. 2014.

[17] Martin Bachratý and J. Širáň. Polarity graphs revisited. *Ars Mathematica Contemporanea*, 8:55–67, 2015.

[18] Kevin J. Barker et al. On the feasibility of optical circuit switching for high performance computing systems. In *Proc. Int. Conf. High Performance Computing, Networking, Storage and Analysis*, pages 1–22, 2005.

[19] Alain Barrat and Martin Weigt. On the properties of small-world network models. *European Physical Journal B-Condensed Matter and Complex Systems*, 13(3):547–560, 2000.

[20] Roberto Battiti and Giampietro Tecchiolli. The reactive tabu search. *ORSA J. Computing*, 6(2):126–140, 1994.

[21] 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*, pages 348–359, Nov. 2014.

[22] Albrecht Beutelspacher and Ute Rosenbaum. *Projective Geometry: From Foundations to Applications*. Cambridge University Press, 1998.

[23] 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.*, 320:155–174, Jun. 2004.

[24] N. I. Biggs. *Algebraic Graph Theory*. Cambridge University Press, second edition, 1993.

[25] Arthur S Bland, Jack C Wells, Otis E Messer, Oscar R Hernandez, and James H Rogers. Titan: Early experience with the Cray XK6 at Oak Ridge National Laboratory. In *Proc. Cray User Group Conf.*, May 2012.

[26] Laurent Bobelin, Arnaud Legrand, M'arquez David, Pierre Navarro, Martin Quinson, Fr'ed'eric Sutar, and Christophe Thiery. Scalable multi-purpose network representation for large scale distributed system simulation. In *Proc. IEEE/ACM Int. Symp. Cluster, Cloud and Grid Computing*, pages 220–227, May 2012.

[27] Béla Bollobás. *Random Graphs (Second Edition)*. Cambridge University Press, 2001.

[28] Béla Bollobás and F. R. K.Chung. The diameter of a cycle plus a random matching. *SIAM J. Discrete Math.*, 1:328–333, 1988.

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

[30] Jesus Camacho and Jose Flich. HPC-Mesh: A homogeneous parallel concentrated mesh for fault-tolerance and energy savings. In *Proc. ACM/IEEE Symp. Archit. Networking Commun. Syst.*, pages 69–80, Oct. 2011.

[31] Cristóbal Camarero, Carmen Martínez, Enrique Vallejo, and Ramón Beivide. Projective networks: Topologies for large parallel computer systems. *IEEE Trans. Parallel Distrib. Syst.*, 28(7):2003–2016, 2017.

[32] Henri Casanova, Arnaud Giersch, Arnaud Legrand, Martin Quinson, and Frédéric Suter. Versatile, scalable, and accurate simulation of distributed applications and platforms. *J. Parallel Distrib. Computing*, 74(10):2899–2917, June 2014.

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

[34] Reuven Cohen and Shlomo Havlin. Scale-free networks are ultrasmall. *Physical Review Letters*, 90, Feb. 2003.

[35] David T Connolly. An improved annealing scheme for the QAP. *European J. Operational Research*, 46(1):93–100, 1990.

[36] Andrew R. Curtis, Tommy Carpenter, Mustafa Elsheikh, Alejandro López-Ortiz, and S. Keshav. REWIRE: An optimization-based framework for unstructured data center network design. In *Proc. Int. Conf. Computer Communications (INFOCOM)*, pages 1116–1124, Mar. 2012.

[37] William J Dally. Performance analysis of k-ary n-cube interconnection networks. *IEEE Trans. Comput.*, 39:775–785, Jun. 1990.

[38] William J Dally and Charles L Seitz. Deadlock-free message routing in multiprocessor interconnection networks. *IEEE Trans. Comput.*, C-36:547–553, May 1987.

[39] William James Dally and Brian Patrick Towles. *Principles and Practices of Interconnection Networks*. Morgan Kaufmann, 2004.

[40] Reetuparna Das, Satish Narayanasamy, Sudhir K Satpathy, and Ronald G Dreslinski. Catnap: Energy proportional multiple network-on-chip. In *Proc. Int. Symp. Comput. Archit.*, pages 320–331, Jun. 2013.

[41] Sourav Das, Dongjin Lee, Dae Hyun Kim, and Partha Pratim Pande. Small-world network enabled energy efficient and robust 3d noc architectures. In *Proc. of the 25th edition on Great Lakes Symposium on VLSI*, pages 133–138, May 2015.

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

[43] Jeffrey H. Dinitz and Douglas R. Stinson. *Contemporary Design Theory: A Collection of Surveys*. John Wiley & Sons, 1992.

[44] Ron O. Dror et al. Overcoming communication latency barriers in massively parallel scientific computation. *IEEE Micro*, 31(3):8–19, 2011.

[45] Masoumeh Ebrahimi and Masoud Daneshtalab. EbDa: a new theory on design and verification of deadlock-free interconnection networks. In *Proc. Int. Symp. Comput. Archit.*, pages 703–715, Jun. 2017.

[46] Kemel Efe and Antonio Fernández. Products of networks with logarithmic diameter and fixed degree. *IEEE Trans. Parallel Distrib. Syst.*, 6:963–975, Sep. 1995.

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

[48] Bernard Elspas. Topological constraints on interconnection-limited logic. In *Proc. Symp. Switching Circuit Theory Logical Design*, pages 133–137, Nov. 1964.

[49] Paul Erdős and Alfréd Rényi. On random graphs I. *Publicationes Mathematicae (Debrecen)*, 6:290–297, 1959.

[50] Paul Erdős and Alfréd Rényi. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci*, 5(1):17–60, 1960.

[51] Peyman Faizian, Md Atiqul Mollah, Xin Yuan, Zaid Alzaid, Scott Pakin, and Michael Lang. Random regular graph and generalized de Bruijn graph with k-shortest path routing. *IEEE Trans. Parallel Distrib. Syst.*, 29(1):144–155, 2018.

[52] Reza Zanjirani Farahani and Masoud Hekmatfar. *Facility location: concepts, models, algorithms and case studies*. Springer, 2009.

[53] Walter Feit and Graham Higman. The nonexistence of certain generalized polygons. *J. algebra*, 1(2):114–131, 1964.

[54] Sonja Filiposka and Carlos Juiz. Community-based complex cloud data center. *Physica A: Statistical Mechanics and its Applications*, 419:356–372, 2015.

[55] José Flich, Tor Skeie, Andrés Mejía, Olav Lysne, Pedro López, Antonio Robles, José Duato, Michihiro Koibuchi, Tomas Rokicki, and José Carlos Sancho. A survey and evaluation of topology-agnostic deterministic routing algorithms. *IEEE Trans. Parallel Distrib. Syst.*, 23:405–425, 2012.

[56] Satoshi Fujita, Koji Nakano, Michihiro Koibuchi, and Ikki Fujiwara. Deterministic construction of regular geometric graphs with short average distance and limited edge length. In *Proc. Int. Conf. Algorithms Archit. Parallel Process.*, pages 295–309, Dec. 2016.

[57] Ikki Fujiwara, Michihiro Koibuchi, and Henri Casanova. Cabinet layout optimization of supercomputer topologies for shorter cable length. In *Proc. Int. Conf. Parallel Distrib. Comput. Appl. Technol.*, pages 227–232, Dec. 2012.

[58] M.R. Garey and D.S. Johnson. Some simplified NP-complete graph problems. *Theoretical Comput. Sci.*, 1:237–267, Feb. 1976.

[59] Christopher J. Glass and Lionel M. Ni. The turn model for adaptive routing. In *Proc. of the Int'l Symp. on Computer Architecture*, pages 441–450, May 1992.

[60] Chuanxiong Guo, Guohan Lu, Dan Li, Haitao Wu, Xuan Zhang, Yunfeng Shi, Chen Tian, Yongguang Zhang, and Songwu Lu. BCube: a high performance, server-centric network architecture for modular data centers. *ACM SIGCOMM Conf. Data Commun.*, pages 63–74, Aug. 2009.

[61] Chuanxiong Guo, Haitao Wu, Kun Tan, Lei Shi, Yongguang Zhang, and Songwu Lu. DCell: A scalable and fault-tolerant network structure for data centers. In *ACM SIGCOMM Conf. Data Commun.*, pages 75–86, Aug. 2008.

[62] Lásló Gyarmati and Tuan Anh Trinh. Scafida: A scale-free network inspired data center architecture. *ACM SIGCOMM Computer Communication Review*, 40:5–12, Oct. 2010.

[63] W. Daniel Hillis and Lewis W. Tucker. The CM-5 connection machine: A scalable supercomputer. *Commun. ACM*, 36, Jan. 1993.

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

[65] George Karypis, Rajat Aggarwal, Vipin Kumar, and Shashi Shekhar. Multilevel hypergraph partitioning: Applications in VLSI domain. In *Proc. Design Automation Conf.*, pages 526–529, Jun. 1997.

[66] George Karypis and Vipin Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM J. Scientific Computing*, 20:359–392, Aug. 1998.

[67] Georgios Kathareios, Cyriel Minkenberg, Bogdan Prisacari, German Rodriguez, and Torsten Hoefler. Cost-effective diameter-two topologies: Analysis and evaluation. In *Proc. Int. Conf. High Performance Computing, Networking, Storage and Analysis*, pages 36:1–36:11, 2015.

[68] Ryuta Kawano, Seiichi Tade, Ikki Fujiwara, Hiroki Matsutani, Hideharu Amano, and Michihiro Koibuchi. Optimized core-links for low-latency nocs. In *Proc. Euromicro Int. Conf. Parallel Distrib. Network-Based Process.*, pages 172–176, Mar. 2015.

[69] Ryuta Kawano, Ryota Yasudo, Hiroki Matsutani, Michihiro Koibuchi, and Hideharu Amano. HiRy: An advanced theory on design of deadlock-free adaptive routing for arbitrary topologies. In *Proc. Int. Conf. Parallel Distrib. Syst.*, pages 664–673, Dec. 2017.

[70] John Kim, William J. Dally, and Dennis Abts. Flattened butterfly: A cost-efficient topology for high-radix networks. In *Proc. Int. Symp. Comput. Archit.*, pages 126–137, Jun. 2007.

[71] John Kim, William J. Dally, Steve Scott, and Dennis Abts. Technology-driven, highly-scalable dragonfly topology. In *Proc. Int. Symp. Comput. Archit.*, pages 77–88, Jun. 2008.

[72] John Kim, William J Dally, Brian Towles, and Amit K Gupta. Microarchitecture of a high radix router. In *Proc. Int. Symp. Comput. Archit.*, pages 420–431, 2005.

[73] S. Kirkpatrick, C. D. Gelatt Jr., and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, May 1983.

[74] Michihiro Koibuchi, Ikki Fujiwara, Hiroki Matsutani, and Henri Casanova. Layout-conscious random topologies for HPC off-chip interconnects. In *Proc. Int. Symp. High Performance Comput. Archit.*, pages 484–495, Feb. 2013.

[75] Michihiro Koibuchi, Akira Funahashi, Akiya Jouraku, and Hideharu Amano. L-turn routing: An adaptive routing in irregular networks. In *Int. Conf. Parallel Process.*, pages 383–392, 2001.

[76] Michihiro Koibuchi, Hiroki Matsutani, Hideharu Amano, D. Frank Hsu, and Henri Casanova. A case for random shortcut topologies for HPC interconnects. In *Proc. Int. Symp. Comput. Archit.*, pages 177–188, Jun. 2012.

[77] Fei Lei, Dezun Dong, Xiangke Liao, Xing Su, and Cunlu Li. Galaxyfly: A novel family of flexible-radix low-diameter topologies for large-scales interconnection networks. In *Proc. Int. Conf. Supercomputing*, pages 1–12, Jun. 2016.

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

[79] F. W. Levi. *Finite Geometrical Systems*. University of Calcutta, 1942.

[80] Yamin Li, Shietung Peng, and Wanming Chu. Recursive dual-net: A new universal network for supercomputers of the next generation. In *Int. Conf. Algorithms and Archit. for Parallel Process.*, pages 809–820, 2009.

[81] Xiang-Ke Liao, Zheng-Bin Pang, Ke-Fei Wang, Yu-Tong Lu, Min Xie, Jun Xia, De-Zun Dong, and Guang Suo. High performance interconnect network for Tianhe system. *J. Comput. Sci. Technol.*, 30, Mar. 2015.

[82] Hiroki Matsutani, Michihiro Koibuchi, Yutaka Yamada, D Frank Hsu, and Hideharu Amano. Fat h-tree: A cost-efficient tree-based on-chip network. 20(8):1126–1141, 2009.

[83] B. D. McKay, M. Miller, and J. Širáň. A note on large graphs of diameter two and given maximum degree. *J. Combinatorial Theory, Series B*, 74:110–118, 1998.

[84] Peter Merz and Bernd Freisleben. A comparison of memetic algorithms, tabu search, and ant colonies for the quadratic assignment problem. In *Proc. Congress on Evolutionary Computation*, volume 3, pages 2063–2070, 1999.

[85] Stanley Milgram. The small world problem. *Psycology Today*, 2, 1967.

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

[87] Ryosuke Mizuno and Yawara Ishida. Constructing large-scale low-latency network from small optimal networks. In *Proc. Int. Symp. Netw.-on-Chip*, pages 1–6, Sep. 2016.

[88] Mehrnaz Moudi and Mohamed Othman. The challenge of interconnect topologies to improve communication in supercomputers. In *Proc. Int. Conf. Recent Trends Inform. Process. Computing*, pages 137–144, Dec. 2012.

[89] Hiroshi Nakahara, Ng. Anh Vu Doan, Ryota Yasudo, and Hideharu Amano. XYZ-randomization using TSVs for low-latency energy efficient 3D-NoCs. In *Proc. Int. Symp. Networks-on-Chip*, pages 1–8, 2017.

[90] 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.*, pages 340–349, Aug. 2016.

[91] Koji Nakano. Linear layouts of generalized hypercubes. *Int. J. Foundations Comput. Sci.*, 14:137–156, Feb. 2003.

[92] Dongkyung Nam and Cheol Hoon Park. Multiobjective simulated annealing: A comparative study to evolutionary algorithms. *Int. J. Fuzzy Systems*, 2(2):87–97, 2000.

[93] M. E. J. Newman, S. H. Strogatz, and D. J. Watts. Random graphs with arbitrary degree distributions and their applications. *Physical Review E*, 64, July 2001.

[94] Mark E. J. Newman, Christopher Moore, and Duncan J. Watts. Mean-field solution of the small-world network model. *Physical Review Letters*, 84(14):3201–3204, 2000.

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

[96] L. Pardalos and M. Resende. A greedy randomized adaptive search procedure for the quadratic assignment problem. *DIMACS Series on Discrete Mathematics and Theoretical Computer Science*, 16:237–261, 1994.

[97] Aniruddh Ramrakhyani, Paul V Gratz, and Tushar Krishna. Synchronized Progress in Interconnection Networks (SPIN): A new theory for deadlock freedom. Jun. 2018.

[98] Lawrence G. Roberts and Barry D. Wessler. Computer network development to achieve resource sharing. In *Proc. Spring Joint Comput. Conf.*, pages 543–549, May 1970.

[99] Sartaj Sahni and Teofilo Gonzalez. P-complete approximation problems. *J. ACM*, 23(3):555–565, 1976.

[100] José Carlos Sancho, Antonio Robles, and José Duato. A new methodology to compute deadlock-free routing tables for irregular networks. In *Int. Workshop Communication, Architecture, and Applications for Network-Based Parallel Computing*, pages 45–60, 2000.

[101] José Carlos Sancho, Antonio Robles, and José Duato. An effective methodology to improve the performance of the up*/down* routing algorithm. *IEEE Trans. Parallel Distrib. Syst.*, 15(8):740–754, 2004.

[102] José Carlos Sancho, Antonio Robles, José Flich, Pedro Lopez, and José Duato. Effective methodology for deadlock-free minimal routing in infiniband networks. In *Proc. International Conf. Parallel Process.*, pages 409–418, 2002.

[103] Michael D. Schroeder et al. Autonet: A high-speed, self-configuring local area network using point-to-point links. *J. Selected Areas Commun.*, 9(8):1318–1335, 1991.

[104] Steve Scott, Dennis Abts, John Kim, and William J. Dally. The BlackWidow High-Radix Clos Network. In *Proc. Int. Symp. Comput. Archit.*, pages 16–28, 2006.

[105] R. Seifert. *Gigabit Ethernet: Technology and Applications for High-Speed LANs*. Addison-Wesley Longman Publishing Co., Inc., 1998.

[106] Charles L. Seitz. The cosmic cube. *Commun. ACM*, 28, Jan. 1985.

[107] Nobutaka Shimizu and Ryuhei Mori. Average shortest path length of graphs of diameter 3. In *Proc. Int. Symp. Netw.-on-Chip*, pages 1–6, Sept. 2016.

[108] Ji-Yong Shin, Bernard Wong, and Emin Gün Sirer. Small-world datacenters. In *Proc. ACM Symp. Cloud Computing*, pages 1–13, 2011.

[109] Ankit Singla, P. Brighten Godfrey, and Alexandra Kolla. High throughput data center topology design. In *Proc. of the 9th USENIX conference on Networked Systems Design and Implementation*, pages 29–41, Apr. 2014.

[110] Ankit Singla, Chi-Yao Hong, Lucian Popa, and P. Brighten Godfrey. Jellyfish: Networking data centers randomly. In *Proc. USENIX Conf. Networked Syst. Design and Implementation*, pages 17:1–17:14, Apr. 2012.

[111] Robert Singleton. On minimal graphs of maximum even girth. *J. Combinatorial Theory*, 1(3):306–332, 1966.

[112] Tor Skeie, Olav Lysne, Jose Flich, Pedro Lopez, Antonio Robles, and Jose Duato. LASH-TOR: A generic transition-oriented routing algorithm. In *Proc. Int. Conf. Parallel Distrib. Syst.*, pages 595–604, Jul. 2004.

[113] Tor Skeie, Olav Lysne, and Ingebjørg Theiss. Layered shortest path (LASH) routing in irregular system area networks. In *Proc. Int. Parallel Distrib. Process. Symp.*, pages 1–8, 2002.

[114] Ray Solomonoff and Anatol Rapoport. Connectivity of random nets. *The bulletin of mathematical biophysics*, 13(2):107–117, 1951.

[115] Shinji Sumimoto, Kazuichi Ooe, Kouichi Kumon, Taisuke Boku, Mitsuhisa Sato, and Akira Ukawa. A scalable communication layer for multi-dimensional hyper crossbar network using multiple gigabit ethernet. In *Proc. Int. Conf. Supercomputing*, pages 107–115, Jun. 2006.

[116] E. D. Taillard. Fant: Fast ant system. *Technical Report*, 1998.

[117] Éric Taillard. Robust taboo search for the quadratic assignment problem. *Parallel computing*, 17(4-5):443–455, 1991.

[118] G. Della Vecchia and C. Sanges. A recursively scalable network VLSI implementation. *Future Generation Computer Systems*, 4(3):235–243, 1988.

[119] Joe H Ward Jr. Hierarchical grouping to optimize an objective function. *J. American Statistical Association*, 58(301):236–244, 1963.

[120] Takafumi Watanabe, Masahiro Nakao, Tomoyuki Hiroyasu, Tomohiro Otsuka, and Michihiro Koibuchi. Impact of topology and link aggregation on a PC cluster with Ethernet. In *Int. Conf. Cluster Comput.*, pages 280–285, Sept. 2008.

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

[122] Michael Woodacre, Derek Robb, Dean Roe, and Karl Feind. The SGI Altix 3000 global shared-memory architecture. In *SGI white paper*.

[123] Haofan Yang, Jyoti Tripathi, Natalie Enright Jerger, and Dan Gibson. Dodec: random-link, low-radix on-chip networks. In *Proc. Int. Symp. Microarchitecture*, pages 496–508, Dec. 2014.

[124] Ryota Yasudo, Michihiro Koibuchi, Koji Nakano, Hiroki Matsutani, and Hideharu Amano. Order/radix problem: Towards low end-to-end latency interconnection networks. In *Proc. Int. Conf. Parallel Process.*, pages 322–331, Aug. 2017.

Publications

Related Papers

Journal Papers

[1] Ryota Yasudo, Michihiro Koibuchi, Koji Nakano, Hiroki Matsutani, and Hideharu Amano. Designing High-Performance Interconnection Networks with Host-Switch Graphs, *IEEE Transactions on Parallel and Distributed Systems*, 2018. (accepted)

International Conference Papers

[2] Ryota Yasudo, Michihiro Koibuchi, Koji Nakano, Hiroki Matsutani, and Hideharu Amano. Order/Radix Problem: Towards Low End-to-End Latency Interconnection Networks, In *Proc. of the 46th International Conference on Parallel Processing*, pp.322–331, Bristol, United Kingdom, August 2017.

Domestic Conference Papers and Technical Reports

[3] Ryota Yasudo, Michihiro Koibuchi, Hideharu Amano, and Koji Nakano. Theoretical Model of Interconnection Networks Consisting of Hosts and Switches, *IEICE Technical Reports*, vol. 116, no. 381, COMP2016-40, pp. 51–58, December 2016.

Other Papers

Journal Papers

[4] Ryota Yasudo, Hiroki Matsutani, Michihiro Koibuchi, Hideharu Amano, and Tadao Nakamura. Scalable Networks-on-Chip with Elastic Links Demarcated by Decentralized Routers, *IEEE Transactions on Computers*, vol. 66, no. 4, pp. 702–716, April 2017. **(TELECOM System Technology Student Award 2017, IEEE Computer Society Japan Chapter Young Author Award 2018)**

International Conference Papers

[5] Ryota Yasudo, Jose Gabriel Figueiredo Continho, Ana Lucia Varbanescu, Wayne Luk, Hideharu Amano, and Tobias Becker. Performance Estimation for Exascale Reconfigurable Dataflow Platforms, In *Proc. of the International Conference on Field-Programmable Technology*, Naha, Japan, December 2018. (to appear)

[6] Ryuta Kawano, Ryota Yasudo, Hiroki Matsutani, and Hideharu Amano. k-Optimized Path Routing for High-Throughput Data Center Networks, In *Proc. of the Sixth International Symposium on Computing and Networking*, Hida Takayama, Japan, November 2018. **(CANDAR'18 Outstanding Paper)**

[7] Ryota Yasudo, Ana Lucia Varbanescu, Jose Gabriel Figueiredo Coutinho, Wayne Luk, and Hideharu Amano. Performance Prediction for Large-scale Heterogeneous Platforms, In *Proc. of the 26th IEEE International Symposium on Field-Programmable Custom Computing Machines*, Boulder, CO, USA, April/May 2018.

[8] Ryuta Kawano, Ryota Yasudo, Hiroki Matsutani, Michihiro Koibuchi, and Hideharu Amano. HiRy: An Advanced Theory on Design of Deadlock-free Adaptive Routing for Arbitrary Topologies, In *Proc. of the IEEE 23rd International Conference on Parallel and Distributed Systems*, pp. 664–673, Shenzhen, China, December 2017.

[9] Hiroshi Nakahara, Nguyen Anh Vu Doan, Ryota Yasudo, and Hideharu Amano. XYZ-Randomization using TSVs for Low-Latency Energy-Efficient 3D-NoCs, In *Proc. of the 11th International Symposium on Networks-on-Chip*, Article no. 17, Seoul, South Korea, October 2017.

[10] Hiroshi Nakahara, Ryota Yasudo, Hiroki Matsutani, Hideharu Amano, and Michihiro Koibuchi. 3D layout of Spidergon, Flattened Butterfly and Dragonfly on a chip stack with inductive coupling through chip interface, In *Proc. of the 14th International Symposium on Pervasive Systems, Algorithms, and Networks, IEEE Computer Society Press*, pp.52–59, Exeter, Devon, United Kingdom, June 2017.

[11] Ryota Yasudo, Hiroki Matsutani, Michihiro Koibuchi, Hideharu Amano, and Tadao Nakamura. On-Chip Decentralized Routers with Balanced Pipelines for Avoiding Interconnect Bottleneck, In *Proc. of the 9th ACM/IEEE International Symposium on Networks-on-Chip*, Article No. 16, pp.1–8, Vancouver, BC, Canada, September 2015.

[12] Ryota Yasudo, Takahiro Kagami, Hideharu Amano, Yasunobu Nakase, Masashi Watanabe, Tsukasa Oishi, Toru Shimizu, and Tadao Nakamura. Design of a Low Power NoC Router Using Marching Memory Through Type, In *Proc. of the 8th IEEE/ACM International Symposium on Networks-on-Chip*, pp.111–118, Ferrara, Italy, September 2014.

[13] Ryota Yasudo, Takahiro Kagami, Hideharu Amano, Yasunobu Nakase, Masashi Watanebe, Tsukasa Oishi, Toru Shimizu, and Tadao Nakamura. A low power NoC router using the

marching memory through type, In *Proc. of the17th IEEE Symposium on Low-Power and High-Speed Chips*, pp.1–3, Yokohama, Japan, April 2014.

Domestic Conference Papers and Technical Reports

[14] Ryuta Kawano, Ryota Yasudo, Hiroki Matsutani, Michihiro Koibuchi, and Hideharu Amano. A Scalable Multi-Path Selection Method for High-Throughput Interconnection Networks, *IEICE Technical Reports*, vol. 118, no. 339, CPSY2018-38, pp. 11–16, December 2018. (In Japanese)

[15] Ryuta Kawano, Ryota Yasudo, Hiroki Matsutani, Michihiro Koibuchi, and Hideharu Amano. Measuring and Understanding Throughput of Routing Algorithms, *IEICE Technical Reports*, vol. 118, no. 165, CPSY2018-23, pp. 133–138, July 2018. (In Japanese)

[16] Ryota Yasudo, Hiroki Matsutani, Michihiro Koibuchi, Hideharu Amano, and Tadao Nakamura. An Efficient NoC with Decentralized Routers, *IPSJ SIG Technical Reports*, vol. 2017-ARC-228, no. 6, pp. 1–6, January 2016. (In Japanese) **(IPSJ Yamashita SIG Research Award)**

[17] Hiroshi Nakahara, Daichi Fujiki, Seiichi Tade, Ryota Yasudo, Ryuta Kawano, Hiroki Matsutani, Michihiro Koibuchi, Koji Nakano, and Hideharu Amano. Topology Optimization of 3D-Stacked Chips under Maxiumum Wire Length Constraint, *IEICE Technical Reports*, vol. 115, no. 374, CPSY2015-104, pp. 111–116, December 2015. (In Japanese)

[18] Ryota Yasudo, Hiroki Matsutani, Michihiro Koibuchi, Hideharu Amano, and Tadao Nakamura. A Distributed Router Architecture using transparent latches for Networks-on-Chip, *IEICE Technical Reports*, vol. 114, no. 330, CPSY2014-80, pp. 45–50, November 2014. (In Japanese)

[19] Ryota Yasudo, Takahiro Kagami, Hideharu Amano, Yasunobu Nakase, Masashi Watanebe, Tsukasa Oishi, Toru Shimizu, and Tadao Nakamura. NoC routers using the marching memory through type, *IEICE Technical Reports*, vol. 113, no. 324, CPSY2013-71, pp. 71–76, November 2013. (In Japanese)