1

Introduction

1.1 Motivation

The past decade has witnessed the rapid evolution of a wide range of large-scale computer networks, such as data center networks (DCNs) (Chen et al. 2016; Andreyev 2014), high-performance computers (HPCs) (Trobec et al. 2016), artificial intelligence infrastructures (Gershon et al. 2024), distributed machine learning systems (Yang et al. 2012), artificial intelligence of things (AIOT) systems (Siam et al. 2025), multi-agent networks (MANs) (Oroojlooy and Hajinezhad 2023), and low Earth orbit (LEO) satellite networks (Pan et al. 2023). Such systems are continuously serving as the cornerstones that underlie the proliferation of various network applications and services, unleashing the power of cutting-edge technologies such as artificial intelligence and driving the world toward a more digitalized and intelligent future.

To accommodate the ever-growing network traffic and provide more diversified services with stringent requirements, Cisco Corporation (2020), the crux lies in the design of the large-scale networks that underpin such systems. The design is overwhelmingly complicated as it involves concerns from various aspects, including but not limited to network topology (Xia et al. 2017), routing protocol design (Chen et al. 2016), and traffic engineering mechanism (Shao et al. 2013). Particularly, *network topology*, defined as the *interconnection* relationship among the devices (e.g., switches, servers) in a network, plays a fundamental role that imposes profound impacts on other aspects of the network.

For network topology design, the academia and the industry have devoted years of efforts in developing numerous solutions that are applied to various scenarios and systems. Overall, there are two veins that lead existing works.

Works along the first vein focus on designing concrete topologies for large-scale networks, especially for DCNs (Al-Fares et al. 2008; Walraed-Sullivan et al. 2013; Liu et al. 2013; Singla et al. 2012; Gyarmati and Trinh 2010; Shin et al. 2011; Yu and Qian 2016; Valadarsky et al. 2016) and HPCs (Kim et al. 2008; Ahn et al. 2009;

Shpiner et al. 2017). Such concrete topology designs can be roughly divided into *deterministic* topologies (Al-Fares et al. 2008; Walraed-Sullivan et al. 2013; Liu et al. 2013; Kim et al. 2008; Ahn et al. 2009; Shpiner et al. 2017) and *random* (or *pseudo-random*) topologies (Singla et al. 2012; Gyarmati and Trinh 2010; Shin et al. 2011; Yu and Qian 2016; Valadarsky et al. 2016). The deterministic topologies are mostly designed based on engineering heuristics. Owing to their decent performance and low complexities in deployment and management, such topologies have been widely acknowledged and commonly adopted in practice, with a wealth of follow-on works (e.g., the well-known Fat-Tree topology [Al-Fares et al. 2008] and its variants Facebook Fabric [Andreyev 2014] and Google Jupiter [Singh 2015]). In parallel, some other works adopt tools from graph theory (especially expander graphs [Hoory et al. 2006])[1] to devise topologies that are conducted in a randomized manner in pursuit of better performance (e.g., the average shortest path length and the number of paths between devices). Nonetheless, the applicability of such solutions is hindered by high complexities and prohibitive costs of deployment and management in practice.

Works along the second vein aim to design network topology in a *synthesized* fashion. The key idea is to search topologies by combining engineering insights based on existing topology designs with optimization-based or learning-based approaches. For example, Schlinker et al. (2015) present Condor, an efficient framework that generates candidate topologies subject to pre-specified design objectives and constraints in a declarative manner. Later, Wang et al. (2018) propose an online traffic-driven deep learning solution to generate high-performance network topologies for reconfigurable networks such as optical/wireless networks.

Despite the justified effectiveness of these works, today's network topology design remains an empirical discipline that heavily relies on engineering heuristics and experimental verification. Moreover, the evaluation is often coupled with other technical implementations (e.g., particular routing protocol, traffic engineering, and network administration strategies), making it hard to investigate the merits of the topologies themselves. Unlike some other disciplines in the field of network system optimization (e.g., layered network protocol design) with mature design frameworks (Chiang et al. 2007; Wei et al. 2006; Griffin et al. 2002; Lee et al. 2007), a *systematic* framework to understand, analyze, and evaluate different network topologies, especially for large-scale networks, is still missing. On its way toward being a science, there are two impeding factors centering around the *intrinsic* complexities of topology design and the lack of a *fundamental* understanding of its design space.

On one hand, topology design usually involves various performance metrics that are potentially conflicting with each other, such as scalability, cost, latency,

1 Intuitively, expander graph is a type of large-scale graph with dense connectivity properties.

throughput, and fault-tolerance. On the other hand, large-scale networks contain numerous devices with a massive number of possibilities in their interconnection relationship. Even a minor modification of the interconnections among devices can lead to a significant performance change (Liu et al. 2013). Without a fundamental understanding of the trade-offs among such metrics, topology design remains a costly trial-and-error-based procedure – more like an art than a science. Although existing optimization-based or learning-based approaches are promising to accelerate this procedure, the search space is limited by human heuristics. Besides, the resulting designs still resort to empirical evaluation that only explains *how* they work but not *why* they work, not to mention how they can be improved systematically toward better performances.

To fill such a gap, in this monograph, we present a unified design framework of topology design for large-scale networks, aiming to provide a *first-principled* perspective to understand, analyze, evaluate, and generate topology design in a *systematic* fashion. An overview of the design framework is given in the next section.

1.2 Overview of the Design Framework

In this section, we briefly introduce the three essential parts of the design framework.

1. **A unified design procedure:** Inspired by Shannon's style thinking of *modularization* and *decoupling*, at the core of the framework lies a general design procedure serving as a *unified* language to describe network topology design. Based on extensive reverse-engineering over existing solutions, the procedure provides a *novel* modularized perspective to decouple network topology. Following the procedure, a network topology can be generated step by step in a *top-down* fashion: first by the design of individual modules[2] (aka intra-module design), then by the design of connections that links modules (aka connection design), and finally by the design of how modules are organized via the designed connections (aka inter-module design). Existing topologies can be viewed as instances with different design choices in each step.
2. **Concrete analytic tools for topology design:** For each step in the design procedure, the framework shows dedicated principles to instruct the corresponding design. Under the guidance of such design principles, the framework provides concrete analytic tools based on graph theory and combinatorial design theory. Particularly, the tools of intra-module and

2 In real-world large-scale networks, modules can be server racks, pods, device groups, containers, clusters, etc.

inter-module designs center around the construction of large-scale networks with dense interconnections, while the tools of connection and recursion design aim to achieve an ultra-low latency among modules. Notably, the framework provides a fine-grained distinction of the atomic modes and novel tools to extend connection design. With such tools, the framework not only opens a unified perspective to review existing topologies (e.g., *switch-centric* topologies and *server-centric* topologies in DCNs), but also enables network designers to develop network topologies in a systematic way.

3. **Quantitative analysis of performance metrics:** Based on the two parts and a quantitative analysis upon such design patterns, the framework offers a comprehensive understanding of the trade-offs among the primary performance metrics in topology design. The results justify the effectiveness of existing topology designs, illuminate their positions in the design space, and present an instructive guideline for further improvements.

1.3 Existing Works

In this section, we take a review of existing works on topology design as the preliminaries of our design framework. As mentioned earlier, our review basically follows two veins. One is related to the works of *concrete* designs for large-scale networks. The other vein involves the works that develop tools or frameworks to *search* topology designs subject to pre-specified requirements.

1.3.1 Concrete Topology Designs

Over the past decades, a wealth of concrete topologies have been proposed for large-scale networks, such as DCNs and HPCs. In such networks, part of devices take over computing functions (aka servers, hosts, computers, or terminals), while the other devices manage routing functions (aka switches, routers, or routing devices) (Chen et al. 2016). Depending on the randomness involved in the process of topology design, existing topologies can be divided into *deterministic* topologies and *random* (or *pseudo-random*) topologies. If a topology's design process does not involve any random operation (e.g., the selection of connections between devices), it is called a deterministic topology. Otherwise, it is a random topology.

1.3.1.1 Deterministic Topologies

Regarding deterministic topologies, depending on whether servers are responsible for networking and routing functions (Chen et al. 2016), deterministic topologies can be further divided into *server-centric* topologies and *switch-centric* topologies. If none of the servers are responsible for networking and routing functions in

the topology, it is called a switch-centric topology. Otherwise, it is a server-centric topology.

Regarding *switch-centric* topology design, the representative topologies among existing works are demonstrated as follows.

Fat-Tree (Al-Fares et al. 2008) is an instance of the well-known Clos network topology (Scott et al. 2006). It has been widely adopted in various commercial data center networks. Overall, a Fat-Tree topology is constructed as a multi-layer network using a group of homogeneous commodity switches with the same port number. In particular, when the port number is equal to some even integer p, the topology is called a *p-ary Fat-Tree*. We take the classical three-layer p-ary Fat-Tree as an example. Such a topology contains three layers: edge layer, aggregation layer, and core layer. Switches in the edge and aggregation layers are divided into p *pods*. These pods are interconnected by $p^2/4$ core switches. Each pod contains $p/2$ aggregation switches and $p/2$ edge switches. Each edge switch is directly connected with $p/2$ servers and $p/2$ aggregation switches in the same pod. Meanwhile, each aggregation switch is directly connected with $p/2$ edge switches in the same pod and $p/2$ core switches. Specifically, the i-th aggregation switch in each pod is connected with the $\left\lceil \frac{p}{2}(i-1)+j \right\rceil$-th core switch, for $i, j = 1, 2, 3, ..., p/2$. In total, such a topology can support $p^3/4$ servers with $5p^2/4$ switches. Figure 1.1 shows a three-layer 4-ary Fat-Tree with twenty 4-port switches.

Aspen Tree (Walraed-Sullivan et al. 2013) is an improvement of multi-layer Fat-Tree for better fault-tolerance. We take the Aspen Tree with L layers as an example where $L \geq 3$. Compared to Fat-Tree, it removes links between switches in the l-th layer ($l = 2, 3, ..., L$) and part of switches in the $(l-1)$-th layer from some pods, leaving more redundant links for switches in the remaining pods. Such a

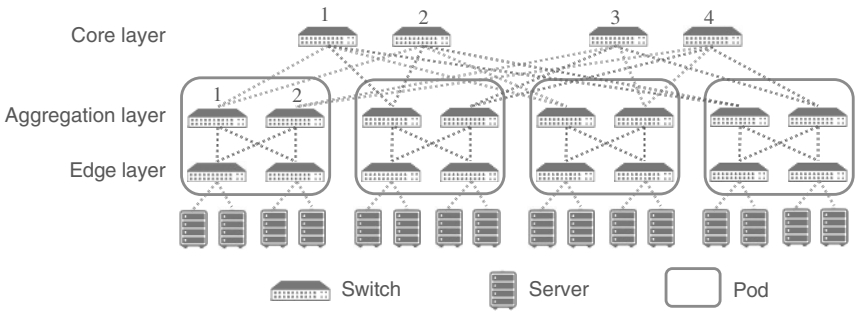

Figure 1.1 An illustration of a three-layer 4-ary Fat-Tree with 4-port switches. This topology contains 16 servers and 20 switches. In each pod, the first aggregation switch is connected to the first and the second core switches, while the second aggregation switch is connected to the third and the fourth core switches.

design improves fault-tolerance at the expense of a lower scalability. Figure 1.2 demonstrates a three-layer Aspen Tree with ten 4-port switches.

AB Fat-Tree (F10) (Liu et al. 2013) is another improvement of multi-layer Fat-Tree for better fault-tolerance. We take the AB Fat-Tree with L layers as an instance where $L \geq 3$. Unlike the homogeneous connection patterns between core switches and pods in Fat-Tree, AB Fat-Tree breaks the symmetry by introducing two distinct connection patterns, called *A-striping* and *B-striping*, respectively. For each $(l-1)$-th-layer switch, it follows the A-striping pattern to be connected with $p/2$ *consecutive* switches in the l-th layer. For each $(l-1)$-th-layer switch, it follows the B-striping pattern to be connected to $p/2$ switches in the l-th layer with a *stride* of $(p/2)^{(l-2)}$. Such a topology leads to a better fault-tolerance than Fat-Tree. Figure 1.3 gives an example of a three-layer AB Fat-Tree with twenty 4-port switches.

HyperX (Ahn et al. 2009) represents a family of topologies, including YARC (Scott et al. 2006), hypercube (Cheng and Chuang 1994), and Flattened Butterfly (Kim et al. 2007). Switches in a level-L HyperX (aka HyperX$_L$) are organized

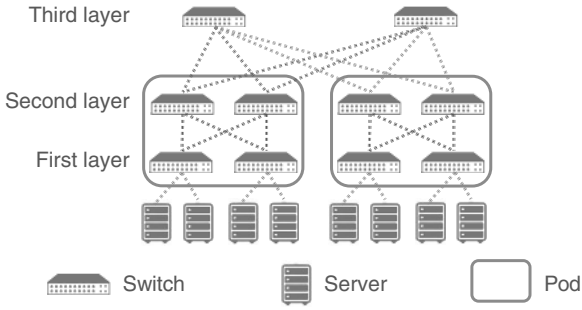

Figure 1.2 An illustration of a three-layer Aspen Tree with ten 4-port switches.

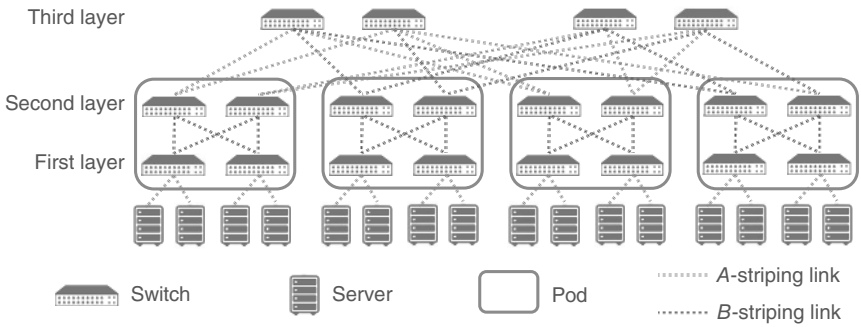

Figure 1.3 An illustration of a three-layer AB Fat-Tree with twenty 4-port switches.

as an L-dimensional integer lattice ($L \geq 1$). Each switch is identified by a coordinate vector $I \triangleq (I_1, I_2, ..., I_L)$ where $I_i = 1, 2, ..., s_i$ for $1 \leq i \leq L$. Such a lattice has a multi-dimensional nested structure. Each i-dimensional lattice contains s_i $(i-1)$-dimensional lattices. Then switches with the same coordinate sub-vector $(I_1, I_2, ..., I_{i-1})$ in all of such $(i-1)$-dimensional lattices are interconnected as a complete graph. Note that a zero-dimensional lattice is a switch. Following this way, HyperX with distinct values of L and s_i leads to different topologies. For example, hypercube can be viewed as an instance of HyperX with $s_i = 2$ for all $1 \leq i \leq L$. Another example is the topology of YARC high-radix switch. It can be viewed as an instance of HyperX with $L = 2$ and $s_1 = s_2 = 8$. Figure 1.4 shows an example of a HyperX with $L = 2$, $s_1 = 2$, and $s_2 = 4$. The topology is formed by eight 5-port switches. Each switch is connected with one and only one server.

As for *server-centric* topology, the typical designs are discussed as follows.

DCell (Guo et al. 2008) is a server-centric topology for DCNs, in which both switches and servers are involved with routing functions. DCell has a recursively-defined hierarchical structure. Each atomic component (called a *level*-0 *DCell* or $DCell_0$) contains p servers which are connected via a shared p-port switch. For each level-i DCell with $i \geq 1$ (aka DCell_i), it is constructed as a complete graph with m_i level-$(i-1)$ DCells. The value of m_i is equal to the number of servers in each level-$(i-1)$ DCell plus one. In this way, a level-i DCell requires each server to be

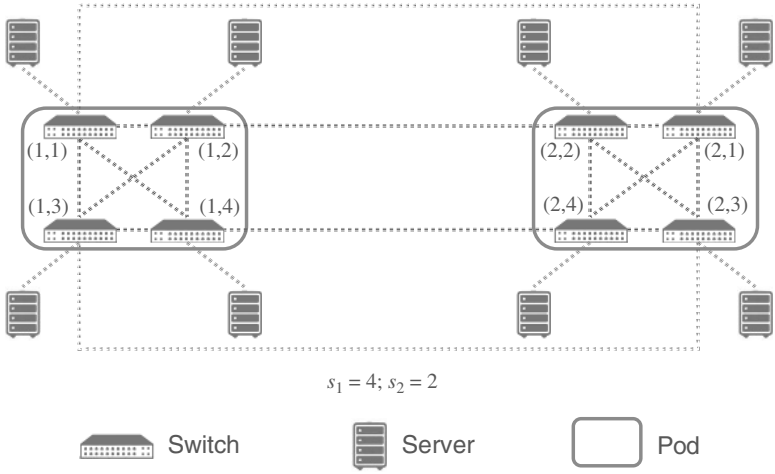

Figure 1.4 An illustration of a level-2 HyperX with eight 5-port switches.

equipped with $(i + 1)$ network interface cards (NICs).[3] Figure 1.5 demonstrates a level-1 DCell with five 4-port switches.

FiConn (Li et al. 2009) is a DCell-like topology. Compared to DCell topology, it requires each server to be equipped with only two NICs (an active NIC and a backup NIC) which makes it applicable for practical use. In such a topology, each atomic component (called a *level*-0 *FiConn* or $FiConn_0$) contains p servers that are connected via a shared p-port switch in the same way as DCell. For each level-i FiConn with $i \geq 1$ (aka FiConn_i), it is constructed as a complete graph with g_i level-$(i-1)$ FiConns. The value of g_i is equal to half of the number of servers with available backup NICs in each level-$(i-1)$ FiConn plus one. Compared with DCell, FiConn incurs lower wiring costs at the expense of a reduced network capacity. Figure 1.6 provides an instance of a level-1 FiConn with three 4-port switches.

BCube (Guo et al. 2009) is another recursively-defined hierarchical topology. Each atomic component (called a *level*-0 *BCube* or $BCube_0$) contains p servers which are connected via a shared p-port switch in the same way as DCell. For each level-i BCube with $i \geq 1$ (aka BCube_i), it is constructed with p level-$(i-1)$ BCubes and p^i switches. Each level-$(i-1)$ BCube contains p^i servers and each server is directly connected to one and only one switch in the level-i BCube. In this way, a level-i BCube requires each server to be equipped with $(i + 1)$ NICs. Figure 1.7 shows a level-1 BCube with eight 4-port switches.

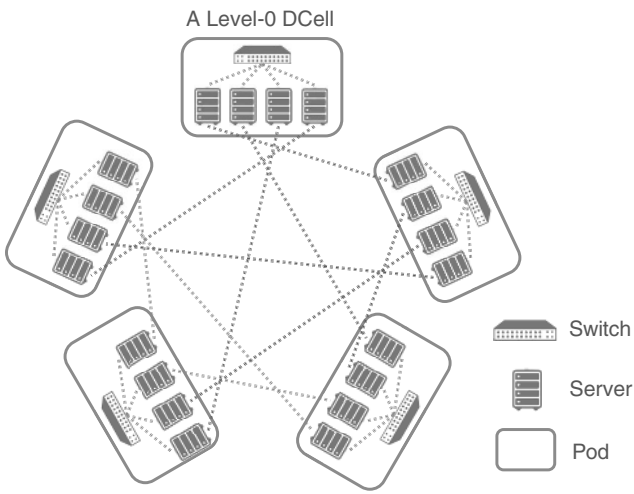

Figure 1.5 An illustration of a level-1 DCell with five 4-port switches.

3 An NIC is a control component of a server via which the server can establish communication links with other devices (e.g., servers and switches) in a network. For each server, the number of its NICs determines the maximum number of devices it can be connected to.

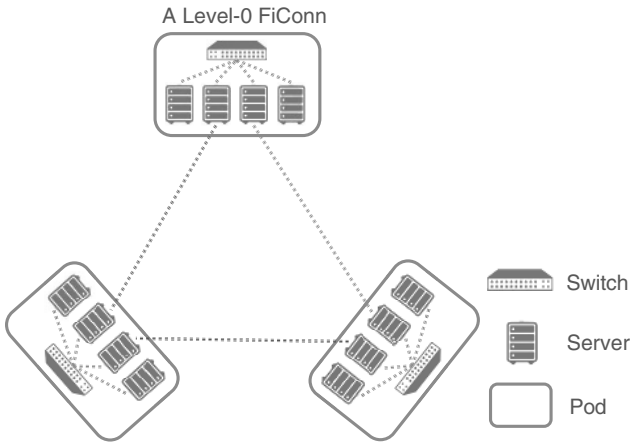

Figure 1.6 An illustration of a level-1 FiConn with three 4-port switches.

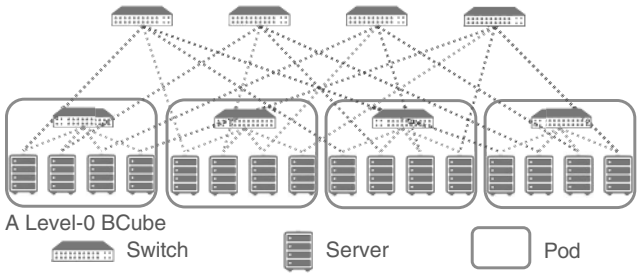

Figure 1.7 An illustration of a level-1 BCube with eight 4-port switches.

1.3.1.2 Random Topologies

Different from deterministic topologies, existing random topologies do not limit the type of routing devices; i.e., they can be either switches or servers with routing functions. Hence, in the following discussion, we do not make the distinction between switch-centric and server-centric designs for random topologies. The representative random topologies among existing works include Jellyfish (Singla et al. 2012), Scafida (Gyarmati and Trinh 2010), Small-World Data Center (Shin et al. 2011), Space Shuffle (Yu and Qian 2016), Xpander (Valadarsky et al. 2016), and Random Folded Clos (Guo and Yang 2015). We provide an exposition for two of them.

Jellyfish (Singla et al. 2012) is a random topology for DCNs by interconnecting switches as an approximately random regular graph. It can be constructed in the following way. Given a group of p-port switches, $(p + 1)$ switches are linked as a

complete graph. For each switch s_i in the remaining switches, it is connected to two switches that are randomly selected from the current topology.[4] Compared with Fat-Tree, on average, Jellyfish's instances can support 25% more servers with a lower latency (Singla et al. 2012). However, it often suffers from a high cabling cost and a complex routing protocol design. Figure 1.8 shows the construction process of a Jellyfish topology with 18 4-port switches.

Xpander (Valadarsky et al. 2016) constructs a topology as an expander graph. Compared with other random topologies, it achieves a higher throughput and a higher scalability. An Xpander topology is composed of several groups of switches. Each group is called a *meta-node*. Each meta-node contains the same number of switches and is connected to all other meta-nodes via the same number of cables. Such a structure facilitates cable bundling among meta-nodes with a lower wiring complexity. Figure 1.9 gives an instance of an Xpander topology with four meta-nodes. Each meta-node contains three switches.

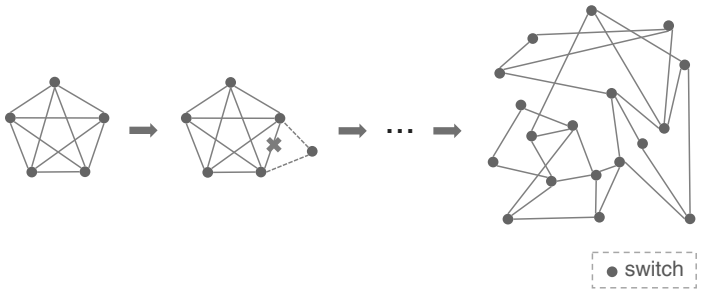

Figure 1.8 The construction process of a Jellyfish topology with 18 4-port switches.

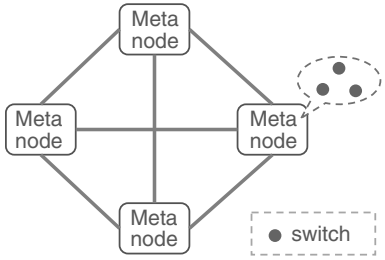

Figure 1.9 An illustration of an Xpander topology with four meta-nodes. Each meta-node contains three switches.

4 This is achieved by removing the link between the two selected switches and reconnecting them with switch s_i.

1.3.2 Search-based Topology Designs

The idea of this vein is to adopt search-based heuristic approaches to generate desired topologies with specific objectives and constraints.[5] To this end, a wealth of frameworks have been proposed in recent years (Schlinker et al. 2015; Curtis et al. 2010, 2012). These frameworks basically take design goals and constraints as inputs and generate a series of candidate topologies (See Figure 1.10). Some classical frameworks are introduced as follows.

LEGUP (Curtis et al. 2010) upgrades existing Clos-like topologies (e.g., Fat-Tree) with a higher bisection bandwidth. Compared to the classical Clos topology with homogeneous switches, LEGUP extends the design with heterogeneous switches and adopts a branch-and-bound algorithm to achieve near-optimal performances (e.g., bisection bandwidth and reliability) under host traffic constraints.

REWIRE (Curtis et al. 2012) is an optimization-based framework to design topologies for DCNs. In particular, REWIRE adopts a simulated annealing-based method to produce topologies that achieve a higher bisection bandwidth and a lower worst-case end-to-end latency than Fat-Tree. However, such a search process does not consider some critical deployment constraints in topology design (e.g., the limitation of physical distance between deployed devices) which may be a concern in practice.

Perseus (Mudigonda et al. 2011) is another optimization-based framework for the design of cost-effective topology in container-sized DCNs. To this end, by considering various primary metrics in topology design (e.g., cabling costs, cable installation costs, switch costs, and hop counts) and their related constraints,

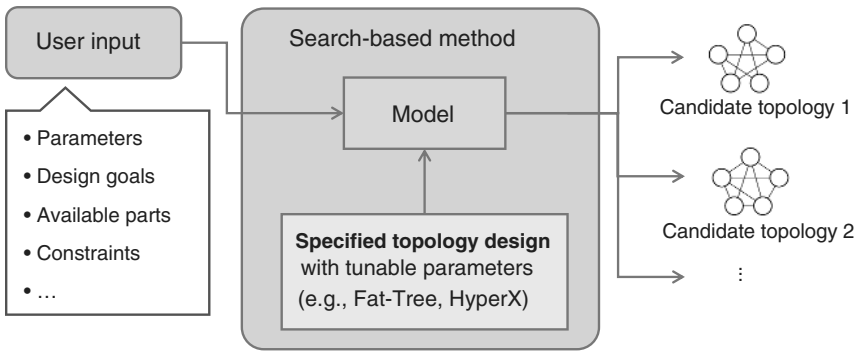

Figure 1.10 An illustration of search-based topology design.

5 Such approaches generally adopt deterministic optimization methods (Curtis et al. 2012) (e.g., linear programming, convex optimization) or machine learning techniques (Wang et al. 2018) (e.g., deep learning).

Perseus provides a workflow to quantify the costs and benefits of potential topologies (especially the extended generalized Fat-Trees [EGFTs] [Ohring et al. 1995] and HyperX topologies), leading designers toward more targeted topology choices.

Condor (Schlinker et al. 2015) is described as a pipeline that facilitates the rapid generation and evaluation of DCN topologies. Compared to concrete topologies, Condor allows designers to express requirements and goals via a declarative topology description language (TDL). According to the TDL descriptions, Condor can synthesize candidate designs with desired performances and support on-demand network expansion based on effective topology evaluations.

xWeaver (Wang et al. 2018) is a traffic-driven deep learning approach which enables high-performance global topology configurations in DCNs. At the core of xWeaver lie a set of properly-structured neural network models. Particularly, by adopting such models to learn the underlying mapping between traffic patterns and topology configurations, xWeaver can generate near-optimal topology configurations with continuous model improvements. Compared to Fat-Tree, xWeaver provides a more flexible and more adaptive way to design topologies that are better suited to ever-changing traffic dynamics.

1.4 Mathematical Background I: Graph Theory

1.4.1 Undirected Graph

An *undirected graph* G is a pair (V, E) such that

- V is a set of nodes;
- E is a set of edges, each of which is an unordered pair of two distinct nodes.

If (u, v) is an edge in an undirected graph, we say that nodes u and v are its *endpoints*. The edge is *incident* with its endpoints. Two nodes are *adjacent* if they are the endpoints of a common edge. In the following examples, we adopt the notation e_{uv} to denote an edge with endpoints u and v.

Example 1.1 *Figure 1.11 shows an example of graphs, where the node set is* $\{1, 2, 3, 4, 5, 6, 7\}$ *and the edge set is* $\{e_{12}, e_{13}, e_{34}, e_{24}, e_{36}, e_{46}, e_{45}, e_{67}, e_{57}\}$. *In such a graph, node* 1 *is incident with edges* e_{12} *and* e_{13}. *Node* 3 *and node* 4 *are the endpoints of the edge* e_{34}. *Node* 4 *and node* 6 *are adjacent.*

1.4.2 Order and Node Degree

For a graph, its *order* is the number of nodes in such a graph. For each node, its *degree* is the number of edges that are incident with it.

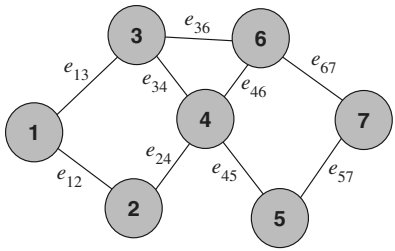

Figure 1.11 An undirected graph with node set V as $\{1, 2, 3, 4, 5, 6, 7\}$ and edge set E as $\{e_{12}, e_{13}, e_{34}, e_{24}, e_{36}, e_{46}, e_{45}, e_{67}, e_{57}\}$.

Example 1.2 *For the graph shown in Figure 1.11, its order is* 7 *and the degree of node* 4 *is* 4.

1.4.3 Path

Given two distinct nodes s and t in a graph $G = (V, E)$, a *path* between them is a sequence of nodes $\langle v_1, v_2, ..., v_k \rangle$ such that $v_1 = s$, $v_k = t$, and $(v_{i-1}, v_i) \in E$ for $i = 2, ..., k$. The *length* of a path is the number of nodes in the path minus one. The *shortest path* between two nodes is the path with the shortest length between them.

Example 1.3 *In the graph shown in Figure 1.11, we select three paths between node* 1 *and node* 7 *as examples, which are denoted by* P_1, P_2 *and* P_3. *The path* P_1 *is* $\langle 1, 3, 6, 7 \rangle$, *the path* P_2 *is* $\langle 1, 2, 4, 5, 7 \rangle$, *and the path* P_3 *is* $\langle 1, 3, 4, 6, 7 \rangle$. *The lengths of paths* P_1, P_2, *and* P_3 *are* 3, 4, *and* 4, *respectively. The shortest path between node* 1 *and node* 7 *is the path* P_1 *with length* 3.

Given two paths $P_1 = \langle v_1, v_2, ..., v_m \rangle$ and $P_2 = \langle u_1, u_2, ..., u_n \rangle$, if $v_1 = u_1$ and $v_m = u_n$, we describe the disjointness of such two paths in the following two ways:

- the paths P_1 and P_2 are *edge-disjoint* if $(v_{i-1}, v_i) \neq (u_{j-1}, u_j)$ for any $1 < i \leq m$ and $1 < j \leq n$.
- the paths P_1 and P_2 are *node-disjoint* if $v_i \neq u_j$ for any $1 < i < m$ and $1 < j < n$.

Example 1.4 *For the paths* P_1, P_2, *and* P_3 *in Example 1.3, the paths* P_1 *and* P_2 *are both edge-disjoint and node-disjoint, while the paths* P_2 *and* P_3 *are edge-disjoint but not node-disjoint. The paths* P_1 *and* P_3 *are neither edge-disjoint nor node-disjoint.*

1.4.4 Distance and Diameter

The *distance* between two nodes is the length of the shortest path between them. For a graph, its *diameter* is the maximum distance between any two nodes in the graph.

Example 1.5 *In the graph shown in Figure 1.11, the distance between node* 1 *and node* 7 *is* 3*. The diameter of the graph shown in Figure 1.11 is* 5*, which is the length of the shortest path between node* 1 *and node* 5*.*

1.4.5 Connectivity

A graph with at least two nodes is *connected* if there is a path between each pair of nodes; otherwise, the graph is *disconnected*. The connectivity of a graph can be described in the following two ways.

- The *edge connectivity* of a graph is the minimum number of edges whose removal will disconnect the graph.
- The *node connectivity* of a graph is the minimum number of nodes whose removal will disconnect the graph.

Example 1.6 *For the graph shown in Figure 1.11, the removal of at least two edges (i.e.,* e_{67} *and* e_{57}*) will disconnect the graph. Hence, its edge connectivity is equal to* 2*.*

Example 1.7 *For the graph shown in Figure 1.11, the removal of at least two nodes (i.e., node* 4 *and node* 6*) will disconnect the graph. Hence, its node connectivity is equal to* 2*.*

In a graph $G = (V, E)$, for any $S \subset V$, we denote its complement as $\bar{S} = V - S$. The set of edges, whose one endpoint belongs to S and another endpoint belongs to $\bar{S}$, is called the *boundary* of S, denoted by ∂S.

Example 1.8 *For the graph shown in Figure 1.11, suppose* $S = \{1, 3, 6\}$*, then its complement* $\bar{S} = \{2, 4, 5, 7\}$ *and its boundary* $\partial S = \{e_{12}, e_{34}, e_{67}, e_{46}\}$*.*

1.4.6 Special Undirected Graphs

Undirected graphs have various classifications based on the properties. In the following parts, the definitions of some special graphs that are involved in this monograph are given.

1.4.6.1 Regular Graph

A *regular graph* is a graph where each node has the same degree. A regular graph with node degree d is called *d-regular graph*.

Example 1.9 *Figure 1.12 shows an example of* 2*-regular graphs.*

1.4.6.2 Complete Graph

Complete graph (CG) is a special case of regular graph where each pair of nodes is adjacent to each other. The complete graph with n nodes is denoted by K_n.

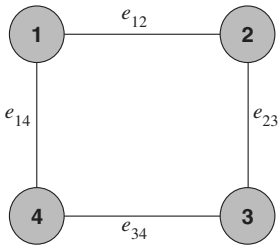

Figure 1.12 A 2-regular graph with node set V as $\{1, 2, 3, 4\}$ and edge set E as $\{e_{12}, e_{23}, e_{34}, e_{14}\}$.

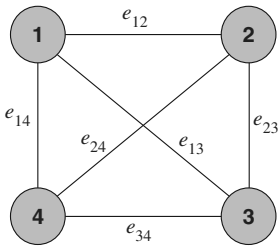

Figure 1.13 A complete graph K_4 (3-regular graph) with node set V as $\{1, 2, 3, 4\}$ and edge set E as $\{e_{12}, e_{23}, e_{34}, e_{14}, e_{13}, e_{24}\}$.

Remark 1.1 A CG K_n is a $(n-1)$-regular graph.

Remark 1.2 The diameter of a CG is *one*.

Example 1.10 *Figure 1.13 shows a CG K_4, which is also a 3-regular graph.*

1.4.6.3 Bipartite Graph

A graph is a *bipartite graph* if its nodes can be partitioned into two disjoint sets V_1 and V_2 (i.e., $V_1 \cap V_2 = \emptyset$), such that any pair of nodes in V_i is not adjacent to each other for $i = 1, 2$. We call each disjoint set as a *node partition*.

Example 1.11 *Figure 1.14 shows an example of bipartite graphs, where one node partition V_1 is* $\{1, 2, 3, 4, 5\}$ *and the other node partition V_2 is* $\{6, 7, 8, 9\}$.

1.4.6.4 Biregular Bipartite Graph

A *biregular bipartite graph* is a bipartite graph where each node in the same node partition has the same degree. Let d_1 and d_2 denote the degree of each node in the two node partitions, respectively. Then the graph is called (d_1, d_2)-*biregular bipartite graph*.

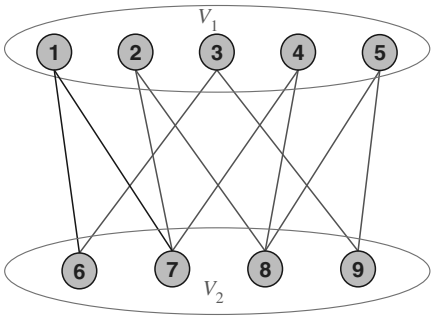

Figure 1.14 A bipartite graph with two node partitions $V_1 = \{1, 2, 3, 4, 5\}$ and $V_2 = \{6, 7, 8, 9\}$.

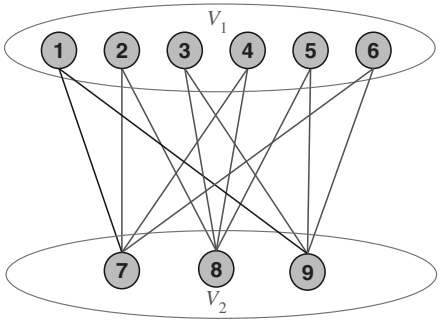

Figure 1.15 A $(2, 4)$-biregular bipartite graph with two partitions of nodes $V_1 = \{1, 2, 3, 4, 5, 6\}$ and $V_2 = \{7, 8, 9\}$. The degree of each node in V_1 is 2 and the degree of each node in V_2 is 4.

Example 1.12 *Figure 1.15 shows an example of biregular bipartite graphs, where one node partition* V_1 *is* $\{1, 2, 3, 4, 5, 6\}$ *and the other node partition* V_2 *is* $\{7, 8, 9\}$. *The degree of each node in* V_1 *is* 2 *and the degree of each node in* V_2 *is* 4.

1.4.6.5 Complete Bipartite Graph

Complete bipartite graph (CBG) is a special case of biregular bipartite graph where each node in one node partition is adjacent to all nodes in the other node partition. Suppose there are m and n nodes in each node partition of a complete bipartite graph, respectively. Then such a complete bipartite graph is denoted by $K_{m,n}$.

Remark 1.3 For a CBG $K_{m,n}$, the degrees of each node in two node partitions are n and m, respectively.

Remark 1.4 The diameter of a CBG is *two*.

Example 1.13 *Figure 1.16 shows an example of complete bipartite graphs* $K_{4,3}$*, where one node partition* V_1 *is* $\{1, 2, 3, 4\}$ *and the other node partition* V_2 *is* $\{5, 6, 7\}$*. The degree of each node in* V_1 *is* 3 *and the degree of each node in* V_2 *is* 4*.*

1.4.6.6 Incidence Graph

Incidence graphs are defined over incidence structures. An *incidence structure* I consists of a set P of points, a set L of lines (disjoint from P), and a relation set $R \subseteq P \times L$. For each pair of point $p \in P$ and line $l \in L$, they are said to be *incident* if $(p, l) \in R$. An *incidence graph* $G(I)$ is defined as a graph with incidence structure I whose node set is formed as $P \cup L$ and edge set contains all edges connecting two incident nodes.

Remark 1.5 An incidence graph is a bipartite graph, where all points and lines are represented by nodes. Particularly, nodes that correspond to points are contained in one partition while nodes that correspond to lines are contained in the other partition. Any two nodes with an incidence relation are connected by an edge.

Example 1.14 *Figure 1.17 shows an incidence graph which contains a point set* $P = \{1, 2, 3\}$*, a line set* $L = \{a, b, c\}$*, and a relation set* $R = \{(1, a), (1, b), (2, b), (2, c), (3, c)\}$*. The two node partitions in such a graph are represented by* V_1 *and* V_2*, respectively.*

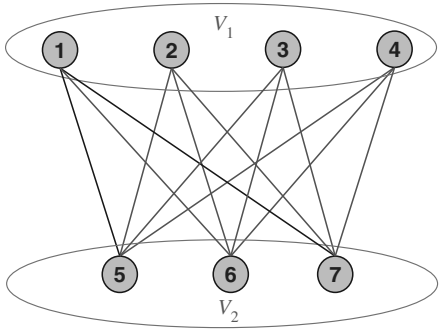

Figure 1.16 A complete bipartite graph $K_{4,3}$ ((3, 4)-biregular bipartite graph) with two node partitions $V_1 = \{1, 2, 3, 4\}$ and $V_2 = \{5, 6, 7\}$.

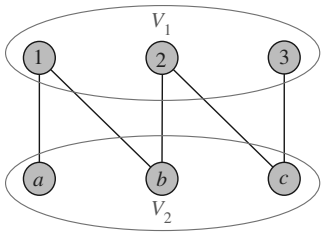

Figure 1.17 An incidence graph with two node partitions $V_1 = \{1, 2, 3\}$ and $V_2 = \{a, b, c\}$.

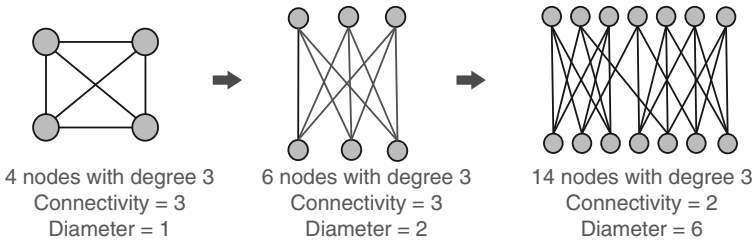

Figure 1.18 Three graphs with different orders given the node degree as three.

1.5 Mathematical Background II: Expander Graph

1.5.1 Expansion Constant

Intuitively, given the degree of each node in a graph, larger order of the graph leads to a tendency toward sparser connectivity and larger diameter. In Figure 1.18, three graphs with node degree as three show such a tendency. Correspondingly, we focus on special large-order graphs with dense connectivity and low diameter. In graph theory, the dense-connectivity property of large-order graphs can be described by expansion property, which is characterized by *expansion constant*. Moreover, graphs with good expansion property can asymptotically achieve the lowest possible diameter.

Definition 1.1 Expansion constant: For a graph $G = (V, E)$ with n nodes, its *expansion constant* $h(G)$ is defined as

$$h(G) = \min_{0<|S|\leq \frac{n}{2}} \frac{|\partial S|}{|S|}, \tag{1.1}$$

where $S \subset V$, $\bar{S} = V - S$, and ∂S denotes the boundary of S.

Example 1.15 *For a complete graph with n nodes, since each node is adjacent to all other nodes in the graph, the boundary of a node sub-set S satisfies* $|\partial S| = |S| * |\bar{S}| = |S| * (n - |S|)$. *Then the expansion constant of the complete graph is* $\frac{n}{2}$ *if n is an even number and* $\frac{n+1}{2}$ *if n is an odd number.*

Remark 1.6 A large value of $h(G)$ implies that there are many edges from any subset of nodes to its complement. Hence, good expansion property can be employed to characterize the dense connectivity of graph G.

Definition 1.2 **Expander graph:** A sequence of d-regular graphs $(G_m)_{m\geq 1}$ of size increasing with m is a family of expander graphs if there is a constant $\epsilon > 0$ such that $h(G_m) \geq \epsilon$ for all i.

Theorem 1.1 *Alon (1986): For a connected d-regular graph with n nodes, its diameter k is* $\Omega(log_{d-1}n)$, *while the diameter of expander graphs is* $O(logn)$.

Remark 1.7 The expander graphs can asymptotically achieve the lowest possible diameter.

1.5.2 Spectra of Graphs

It is known that the problem about determining the expansion constant of arbitrary graph is NP-complete problem (Høholdt and Janwa 2012). Instead, the spectra of graphs can be adopted to estimate the expansion constant because of easy calculability. The related definitions and properties are shown as follows.

Definition 1.3 **Adjacency matrix:** Suppose G is a graph with n nodes, which are indexed by $1, 2, ..., n$, respectively. The adjacency matrix of G is an n-by-n matrix $A(G)$ given by

$$A(G)_{i,j} = \begin{cases} 1\,, \text{ if node } i \text{ and node } j \text{ are adjacent;} \\ 0\,, \text{ otherwise.} \end{cases} \tag{1.2}$$

Example 1.16 *The adjacency matrix of a complete graph with three nodes is*

$$\begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}. \tag{1.3}$$

Definition 1.4 **Spectra of graph:** Let $A(G)$ be the adjacency matrix of a graph G with n nodes. Then $A(G)$ has n real eigenvalues, denoted by $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_n$, which are defined as the spectra of graph G.

Theorem 1.2 *Godsil and Royle (2001): For some special graphs with n nodes, their spectra ($\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_n$) satisfy the following properties.*

- *If G is a d-regular graph, then $\lambda_1 = d$ and $\lambda_n \geq -d$.*
- *If G is a (d_1, d_2)-biregular bipartite graph, then $\lambda_1 = \sqrt{d_1 d_2}$ and $\lambda_n = -\sqrt{d_1 d_2}$.*

Example 1.17 *For any complete graph K_n with n nodes (a special $(n-1)$-regular graph), its spectra are $\lambda_1 = n-1$ and $\lambda_2 = \lambda_3 = ... = \lambda_n = -1$.*

Example 1.18 *For any complete bipartite graph $K_{m,n}$ with m nodes and n nodes in two node partitions (a special (n, m)-biregular bipartite graph), its spectra are $\lambda_1 = \sqrt{mn}$, $\lambda_{m+n} = -\sqrt{mn}$, and $\lambda_2 = \lambda_3 = ...\lambda_{m+n-1} = 0$.*

Definition 1.5 **Nontrivial eigenvalues:** For a connected d-regular graph, any eigenvalue of its adjacency matrix $\lambda_i \neq \pm d$ is called a nontrivial eigenvalue. For a connected (d_1, d_2)-biregular bipartite graph, any eigenvalue of its adjacency matrix $\lambda_i \neq \pm\sqrt{d_1 d_2}$ is called as a nontrivial eigenvalue.

Example 1.19 *By Examples* 1.17*, the nontrivial eigenvalues of K_n include $\lambda_2 = \lambda_3 = ... = \lambda_n = -1$.*

Example 1.20 *By Examples* 1.18*, the nontrivial eigenvalues of $K_{m,n}$ include $\lambda_2 = \lambda_3 = ...\lambda_{m+n-1} = 0$.*

Theorem 1.3 *Alon (1986): Let G be a connected d-regular graph. Then its expansion constant $h(G)$ and the second largest eigenvalue (SLE) λ_2 of its adjacency matrix satisfy*

$$\frac{d-\lambda_2}{2} \leq h(G) \leq \sqrt{2d(d-\lambda_2)}. \tag{1.4}$$

Remark 1.8 For regular graphs, a smaller value of SLE implies a larger value of expansion constant.

1.5.3 Ramanujan Graph

Next, *Ramanujan graph* will be introduced, a special type of graph with good expansion property for regular graph and biregular bipartite graph, respectively. The related definitions and properties are recapped as follows.

Definition 1.6 **Second largest eigenvalue modulus:** For regular graphs and biregular bipartite graphs, their second largest eigenvalue modulus (SLEM) are defined as the largest eigenvalue modulus among all the nontrivial eigenvalues, denoted by $\bar{\lambda}_2$.

Remark 1.9 The second largest eigenvalue modulus (SLEM) $\bar{\lambda}_2$ of graphs satisfies the following properties.

- If G is a d-regular graph with n nodes, then $\bar{\lambda}_2 = max\{|\lambda_2|, |\lambda_n|\} \geq \lambda_2$.
- If G is a (d_1, d_2)-biregular bipartite graph with n nodes, $\bar{\lambda}_2 = max\{|\lambda_2|, |\lambda_{n-1}|\} = \lambda_2$.

Definition 1.7 **Ramanujan graph:** A d-regular graph G is called Ramanujan graph if its second largest eigenvalue modulus $\bar{\lambda}_2$ satisfies

$$\bar{\lambda}_2 \leq 2\sqrt{d-1}. \tag{1.5}$$

Example 1.21 *For any complete graph with n nodes (a special $(n-1)$-regular graph), its spectra are $\lambda_1 = n-1$ and $\lambda_2 = \lambda_3 = ... = \lambda_n = -1$, which implies $\bar{\lambda}_2 = 1$. When $n \geq 3$, we have $\bar{\lambda}_2 = 1 \leq 2\sqrt{(n-1)-1}$. Hence, complete graphs with at least three nodes are Ramanujan graphs.*

Definition 1.8 **Bipartite Ramanujan graph:** A (d_1, d_2)-biregular bipartite graph G is called bipartite Ramanujan graph if its second largest eigenvalue modulus $\bar{\lambda}_2$ satisfies

$$\bar{\lambda}_2 \leq \sqrt{d_1 - 1} + \sqrt{d_2 - 1}. \tag{1.6}$$

Example 1.22 *For any complete bipartite graph with m nodes and n nodes in two partitions of nodes (a special (n, m)-biregular bipartite graph), its SLEM is $\bar{\lambda}_2 = \lambda_2 = 0$. Since $\bar{\lambda}_2 = 0 \leq \sqrt{m-1} + \sqrt{n-1}$, complete bipartite graphs are bipartite Ramanujan graphs.*

Theorem 1.4 *Friedman (2003): Let $\langle G_m = (V_m, E_m)\rangle_{m\geq 1}$ be a family of connected d-regular graphs with $|V_m| \to +\infty$ as $m \to +\infty$. Then*

$$\liminf_{|V_m|\to+\infty} \bar{\lambda}_2 \geq 2\sqrt{d-1}, \tag{1.7}$$

which means the infimum of $\bar{\lambda}_2$ has a horizontal asymptote at $2\sqrt{d-1}$.

Theorem 1.5 *Feng (1996): Let $\langle G_m = (V_m, E_m)\rangle_{m\geq 1}$ be a family of connected (d_1, d_2)-biregular bipartite graphs with $|V_m| \to +\infty$ as $m \to +\infty$. Then*

$$\liminf_{|V_m|\to+\infty} \bar{\lambda}_2 \geq \sqrt{d_1 - 1} + \sqrt{d_2 - 1}, \tag{1.8}$$

which means the infimum of $\bar{\lambda}_2$ has a horizontal asymptote at $\sqrt{d_1 - 1} + \sqrt{d_2 - 1}$.

Remark 1.10 The expansion properties of Ramanujan graphs and bipartite Ramanujan graphs are optimal when the graphs are sufficiently large.

1.6 Mathematical Background III: Combinatorial Design Theory

1.6.1 Overview of Block Design

Combinatorial design theory studies how to select subsets from a finite set to satisfy some specified properties. The members of the finite set are called *elements* and the selected subsets are called *blocks*. If the blocks are simply unordered subsets of elements and there is no structural ordering or pattern between blocks, the set of elements and the collection of blocks constitute a *block design*. Some formal definitions of block designs are given as follows.

Definition 1.9 **Block design:** A *block design* is defined as a pair (X, B), where X is a set of elements and B is a collection (i.e., multi-set) of blocks, such that

- each block is a subset of X (i.e., contains some elements in X),
- there is no structural ordering or pattern between blocks.

Some examples of block design are given as follows. Blocks are described in the form abc instead of $\{a, b, c\}$ for simplicity.

Example 1.23 *An example of block design is given as follows:*

$$\begin{aligned} X &= \{1, 2, 3, 4\}; \\ B &= \{12, 34, 13, 24, 14, 23\}. \end{aligned} \tag{1.9}$$

Remark 1.11 Example 1.23 shows some special properties in block designs, which are listed here:

- *Uniform*. All blocks contain the same number of elements. For Example 1.23, each block contains *two* elements.
- *Regular*: All elements are contained in the same number of blocks. For Example 1.23, each element is contained in *three* blocks.
- *Pairwise balanced*. Each pair of distinct elements is contained in the same number of common blocks. For Example 1.23, each pair of distinct elements is contained in *one* common block.
- *Group divisible*. Elements can be partitioned into some groups, such that each pair of elements is contained in either the same group or the same number of

common blocks. For Example 1.23, elements are partitioned into four groups, i.e., [1], [2], [3], [4]. In this way, each pair of elements in distinct groups is contained in *one* common block.

- *Resolvable*. Blocks can be partitioned into some groups, such that each group of blocks contains each element exactly once. For Example 1.23, blocks are partitioned into three groups, i.e., [12, 34], [13, 24], [14, 23]. In this way, each group of blocks contains each element exactly once.

Remark 1.12 Any pairwise balanced block design is a trivial case of group divisible block design by partitioning elements into singleton groups, i.e., each group consists of exactly one element.

Example 1.24 *An example of block design is given as follows:*

$$\begin{aligned} X &= \{1, 2, 3\}; \\ B &= \{123, 123, 123\}. \end{aligned} \tag{1.10}$$

Example 1.24 is a trivial case of block design called *complete block design*, which is defined as follows.

Definition 1.10 Complete and incomplete block design: A block design is called complete block design (CBD) if each block contains all elements. Otherwise, the block design is called incomplete block design.

Remark 1.13 Any CBD has the uniform property, the regular property, the pairwise balanced property, the group divisible property, and the resolvable property.

Various block designs with different constraints on the properties are simply described in Table 1.1. Next, the most basic block design, i.e., balanced incomplete block design will be introduced first. Then we will show the extended block designs including pairwise balanced design, group divisible design, and transversal design.

Table 1.1 Various block designs.

Block design	Uniform	Regular	Pairwise balanced	Group divisible
Complete block design	•	•	•	•
Balanced incomplete block design	•	•	•	•
Pairwise balanced design	○	○	•	•
Group divisible design	○	○	○	•
Transversal design	•	•	•	•

1.6.2 Balanced Incomplete Block Design

Definition 1.11 **Balanced incomplete block design:** Let v, b, r, k, and λ be positive integers such that $v > k \geq 2$. A balanced incomplete block design-(v, b, r, k, λ) (BIBD-(v, b, r, k, λ)) is a block design (X, B) that satisfies the following properties:

- X is a set of v elements;
- B is a collection of b blocks; each of which contains k elements in X;
- each element is contained in exactly r blocks;
- every pair of distinct elements is contained in exactly λ common blocks.

Remark 1.14 A BIBD is called an *incomplete* block design because $k < v$. A complete block design (CBD) can be regarded as the counterpart of BIBD with $k = v$.

Example 1.25 *Example 1.23 is a BIBD-*(4, 6, 3, 2, 1) *with*

$$\begin{aligned} X &= \{1, 2, 3, 4\}; \\ B &= \{12, 34, 13, 24, 14, 23\}. \end{aligned} \tag{1.11}$$

Example 1.26 *An example of BIBD-*(7, 14, 6, 3, 2) *is given as follows:*

$$\begin{aligned} X &= \{1, 2, 3, 4, 5, 6, 7\}; \\ B &= \{123, 145, 167, 246, 257, 347, 356, \\ &\quad\ 123, 147, 156, 245, 267, 346, 357\}. \end{aligned} \tag{1.12}$$

Definition 1.12 **Resolvable balanced incomplete block design:** A BIBD satisfying resolvable property is called a resolvable balanced incomplete block design (RBIBD). To be specific, b blocks in an RBIBD-(v, b, r, k, λ) can be partitioned into r groups with the same number of blocks, such that each group of blocks contains each element exactly once.

Remark 1.15 For any RBIBD-(v, b, r, k, λ), $b \equiv 0 \ (mod\ r)$.

Example 1.27 *Example 1.23 is a resolvable BIBD-*(4, 6, 3, 2, 1) *with*

$$\begin{aligned} X &= \{1, 2, 3, 4\}; \\ B &= \{[12, 34], [13, 24], [14, 23]\}, \end{aligned} \tag{1.13}$$

in which the notation [ ] *represents the group of blocks such that each group contains all elements exactly once.*

Lemma 1.1 *Stinson (2007): The parameters of a BIBD-(v, b, r, k, λ) satisfy:*

- $r(k-1) = \lambda(v-1)$,
- $vr = bk$,
- $b > r > \lambda$,
- $b \geq v$, $r \geq k$.

Example 1.28 *For the BIBD-$(4, 6, 3, 2, 1)$ in Example 1.23, we have that $v = 4$, $b = 6$, $r = 3$, $k = 2$, and $\lambda = 1$. One can verify that $r(k-1) = \lambda(v-1) = 3$, $vr = bk = 12$, $b = 6 > r = 3 > \lambda = 1$, and $b = 6 \geq v = 4$.*

Remark 1.16 By Lemma 1.1, we have that $r = \dfrac{\lambda(v-1)}{k-1}$ and $b = \dfrac{\lambda v(v-1)}{k(k-1)}$. Hence, a BIBD-$(v, b, r, k, \lambda)$ can also be denoted by BIBD-(v, k, λ) for short.

Corollary 1.1 *The following equations are two necessary conditions for the existence of a BIBD-(v, b, r, k, λ):*

- $\lambda(v-1) \equiv 0 \ (mod\ k-1)$;
- $\lambda v(v-1) \equiv 0 \ (mod\ k(k-1))$.

Remark 1.17 In the following cases, the necessary conditions are also sufficient for the existence of a BIBD-$(v, k, 1)$:

- $k = 2$, $v \geq 2$;
- $k = 3$, $v \equiv 1, 3 \ (mod\ 6)$;
- $k = 4$, $v \equiv 1, 3 \ (mod\ 12)$;
- $k = 5$, $v \equiv 1, 5 \ (mod\ 20)$.

In the following theorems and definitions, some special types of BIBD with existence guarantee will be introduced.

Theorem 1.6 *Stinson (2007): Let B be a collection consisting of all k-subsets of X. Then (X, B) is a BIBD-$(v, k, \binom{v-2}{k-2})$.*

Example 1.29 *An example of the BIBD-$(v, k, \binom{v-2}{k-2})$ with $v = 3$ and $k = 2$ is a BIBD-$(3, 3, 2, 2, 1)$ with*

$$\begin{aligned} &X = \{1, 2, 3\}; \\ &B = \{12, 13, 23\}. \end{aligned} \tag{1.14}$$

Theorem 1.7 *Stinson (2007): When $k = 2$, the BIBD-$(v, k, \binom{v-2}{k-2})$ is resolvable if and only if v is an even number and $v \geq 4$.*

Example 1.30 *An example of the BIBD-*$(v, k, \binom{v-2}{k-2})$ *with* $v = 4$ *and* $k = 2$ *is the resolvable BIBD-*$(4, 6, 3, 2, 1)$ *appeared in Examples 1.23 and 1.27.*

Example 1.31 *An example of the BIBD-*$(v, k, \binom{v-2}{k-2})$ *with* $v = 6$ *and* $k = 2$ *is a resolvable BIBD-*$(6, 15, 5, 2, 1)$ *with*

$$\begin{aligned} X &= \{1, 2, 3, 4, 5, 6\}; \\ B &= \{[12, 34, 56], [13, 25, 46], [14, 26, 35], [15, 24, 36], [16, 23, 45]\}, \end{aligned} \tag{1.15}$$

in which the notation [ ] *represents the group of blocks such that each group contains all elements exactly once.*

Definition 1.13 **Projective plane:** Given an integer $n \geq 2$, a BIBD-$(n^2 + n + 1, n^2 + n + 1, n + 1, n + 1, 1)$ is called a projective plane of order n.

Theorem 1.8 *Kirkman (1850): For every prime power* $n \geq 2$*, there exists a projective plane of order* n*.*

Example 1.32 *A projective plane of order* 2 *is a BIBD-*$(7, 7, 3, 3, 1)$ *with*

$$\begin{aligned} X &= \{1, 2, 3, 4, 5, 6, 7\}; \\ B &= \{123, 145, 167, 246, 257, 347, 356\}. \end{aligned} \tag{1.16}$$

Example 1.33 *A projective plane of order* 4 *is a BIBD-*$(21, 21, 5, 5, 1)$ *with*

$$\begin{aligned} X &= \{A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U\}; \\ B &= \{ABCDE, AFGHI, AJKLM, ANOPQ, ARSTU, BFJNS, BGLOR, \\ &\quad BHMOU, BIKPT, CFKQU, CGMNT, CHLPS, CIJOR, DFMPR, \\ &\quad DGKOS, DHJQT, DILNU, EFLOT, EGJPU, EHKNR, EIMQS\}. \end{aligned} \tag{1.17}$$

Definition 1.14 **Affine plane:** Let $n \geq 2$. A BIBD-$(n^2, n^2 + n, n + 1, n, 1)$ is called an affine plane of order n.

Theorem 1.9 *Kirkman (1850): For every prime power* $n \geq 2$*, there exists an affine plane of order* n*.*

Theorem 1.10 *Bose (1942): Affine plane is a resolvable BIBD.*

Example 1.34 *Example 1.23 is an affine plane of order* 2 *(i.e., BIBD-*(4, 6, 3, 2, 1)*) with*

$$\begin{aligned} &X = \{1, 2, 3, 4\}; \\ &B = \{[12, 34], [13, 24], [14, 23]\}, \end{aligned} \tag{1.18}$$

where the notation [ ] *represents the group of blocks such that each group contains all elements exactly once.*

Example 1.35 *An affine plane of order* 4 *is a BIBD-*(16, 20, 5, 4, 1) *with*

$$\begin{aligned} X = \{&A, B, C, D, E, F, G, H, I, G, K, L, M, N, O, P\}; \\ B = \{&[ABCD, EFGH, IJKL, MNOP], \\ &[AEIN, BGLM, CHJP, DFKO], \\ &[AFLP, BHIO, CGKN, DEJM], \\ &[AHKM, BFJN, CELO, DGIP], \\ &[AGJO, BEKP, CFIM, DHLN]\}, \end{aligned} \tag{1.19}$$

where the notation [ ] *represents the group of blocks such that each group contains all elements exactly once.*

Definition 1.15 **Hadamard matrix:** Let I_n denote the identity matrix of size n. A Hadamard matrix of order n is an n-by-n matrix H in which every entry is 1 or -1 such that $HH^T = nI_n$.

Example 1.36 *A Hadamard matrix of order* 2 *is given by*

$$\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}. \tag{1.20}$$

Example 1.37 *A Hadamard matrix of order* 4 *is given by*

$$\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}. \tag{1.21}$$

Theorem 1.11 *Hadamard (1893): If there exists a Hadamard matrix of order* $n > 2$*, then* $n \equiv 0$ $(mod\ 4)$*.*

Remark 1.18 The smallest order $n \equiv 0$ $(mod\ 4)$ for which a Hadamard matrix is not currently known to exist is $n = 668$.

Theorem 1.12 *Stinson (2007): Suppose $m > 1$. Then there exists a Hadamard matrix of order $4m$ if and only if there exists a BIBD-$(4m-1, 2m-1, m-1)$.*

Example 1.38 *Example 1.32 is a BIBD induced by a Hadamard matrix of order 8 (i.e., BIBD-$(7, 7, 3, 3, 1)$) with*

$$\begin{aligned} &X = \{1, 2, 3, 4, 5, 6, 7\}; \\ &B = \{123, 145, 167, 246, 257, 347, 356\}. \end{aligned} \tag{1.22}$$

Theorem 1.13 *Stinson (2007): Suppose there is a Hadamard matrix of order $4m$. If $m \geq 3$, then there exists a BIBD-$(2m-1, 4m-2, 2m-2, m-1, m-2)$; if $m \geq 2$, then there exists a BIBD-$(2m, 4m-2, 2m-1, m, m-1)$ and such a BIBD is resolvable.*

Example 1.39 *By Theorem 1.13, when $m = 3$, these exists a BIBD-$(5, 10, 4, 2, 1)$ with*

$$\begin{aligned} &X = \{1, 2, 3, 4, 5\}; \\ &B = \{12, 13, 14, 15, 23, 24, 25, 34, 35, 45\}. \end{aligned} \tag{1.23}$$

Example 1.40 *By Theorem 1.13, when $m = 2$, these exists a resolvable BIBD-$(4, 6, 3, 2, 1)$ with*

$$\begin{aligned} &X = \{1, 2, 3, 4\}; \\ &B = \{[12, 34], [13, 24], [14, 23]\}, \end{aligned} \tag{1.24}$$

where the notation [ ] represents the group of blocks such that each group contains all elements exactly once.

By the following theorems and definitions, we present two methods to construct new BIBDs from old ones.

Theorem 1.14 *Stinson (2007): Suppose there exist a BIBD-(v, k, λ_1) and a BIBD-(v, k, λ_2). Then there exists a BIBD-$(v, k, \lambda_1 + \lambda_2)$.*

Example 1.41 *Given a BIBD-$(3, 2, 1)$ parameterized as (X_1, B_1) with*

$$\begin{aligned} &X_1 = \{1, 2, 3\}; \\ &B_1 = \{12, 23, 13\}, \end{aligned} \tag{1.25}$$

and a BIBD-$(3, 2, 2)$ parameterized as (X_2, B_2) with

$$\begin{aligned} &X_2 = \{1, 2, 3\}; \\ &B_2 = \{12, 23, 13, 12, 23, 13\}, \end{aligned} \tag{1.26}$$

the following (X_3, B_3) *is a BIBD-*$(3, 2, 3)$ *with*

$$\begin{aligned} X_3 &= \{1, 2, 3\}; \\ B_3 &= \{12, 23, 13, 12, 23, 13, 12, 23, 13\}. \end{aligned} \tag{1.27}$$

Corollary 1.2 *Suppose there exists a BIBD-*(v, k, λ)*. Then there exists a BIBD-*$(v, k, s\lambda)$ *for any integer* $s \geq 1$.

Definition 1.16 **Complementary design:** Given a BIBD-(v, b, r, k, λ) with element set X and blocks $\{B_1, B_2, ..., B_b\}$, its complementary design is a block design with element set X and blocks $\{X \setminus B_1, X \setminus B_2, ..., , X \setminus B_b\}$.

Example 1.42 *The original BIBD-*$(5, 10, 4, 2, 1)$ *in Example 1.39 is parameterized as* (X_1, B_1) *with*

$$\begin{aligned} X_1 &= \{1, 2, 3, 4, 5\}; \\ B_1 &= \{12, 13, 14, 15, 23, 24, 25, 34, 35, 45\}, \end{aligned} \tag{1.28}$$

and its complementary design is parameterized as (X_2, B_2) *with*

$$\begin{aligned} X_2 &= \{1, 2, 3, 4, 5\}; \\ B_2 &= \{345, 245, 235, 234, 145, 135, 134, 125, 124, 123\}. \end{aligned} \tag{1.29}$$

Theorem 1.15 *Stinson (2007): The complementary design of a BIBD-*(v, b, r, k, λ) *is a BIBD-*$(v, b, b - r, v - k, b - 2r + \lambda)$*, provided that* $b - 2r + \lambda > 0$.

Example 1.43 *The design* (X_2, B_2) *in Example 1.42 is a BIBD-*$(5, 10, 6, 3, 3)$*, which is the complementary design of a BIBD-*$(5, 10, 4, 2, 1)$.

1.6.3 Pairwise Balanced Design and Group Divisible Design

1.6.3.1 Pairwise Balanced Design

Definition 1.17 **Pairwise balanced design:** Suppose $v \geq 2$, $\lambda \geq 1$, and $K \subseteq \{n \in Z : n \geq 2\}$. A pairwise balanced design-(v, K, λ) (PBD-(v, K, λ)) is a block design (X, B) such that the following properties are satisfied:

- X is a set of v elements;
- B is a collection of blocks, each of which is a sub-set of elements in X;
- The number of elements contained in each block belongs to the set K;
- Every pair of distinct elements is contained in exactly λ blocks.

Remark 1.19 A PBD-$(v, K, 1)$ is often simply denoted as a PBD-(v, K).

Example 1.44 *An example of PBD-*$(5, \{2, 3\}, 1)$ *is given as follows:*

$$\begin{aligned} X &= \{1, 2, 3, 4, 5\}; \\ B &= \{125, 134, 23, 24, 35, 45\}. \end{aligned} \tag{1.30}$$

Definition 1.18 **Regular pairwise balanced design:** A PBD-(v, K, λ) is a *regular* pairwise balanced design if each element is contained in exactly r blocks, where r is a positive integer.

Example 1.45 *The following design* (X, B) *is a regular PBD-*$(6, \{2, 3\}, 1)$ *with*

$$\begin{aligned} X &= \{1, 2, 3, 4, 5, 6\}; \\ B &= \{123, 456, 14, 15, 16, 24, 25, 26, 34, 35, 36\}, \end{aligned} \tag{1.31}$$

where each element is contained in exactly four blocks.

Definition 1.19 **Uniform pairwise balanced design:** A PBD-(v, K, λ) is a *uniform* pairwise balanced design if each block contains exactly k elements, i.e., $K = \{k\}$.

Remark 1.20 A complete block design (CBD) is a special case of uniform PBD-$(v, \{k\}, \lambda)$ with $k = v$.

Theorem 1.16 *Stinson (2007): A uniform PBD-*$(v, \{k\}, \lambda)$ *with* $k < v$ *is a BIBD-*(v, k, λ).

Remark 1.21 A uniform PBD is also a regular PBD. However, a regular PBD is not necessarily a uniform PBD.

Theorem 1.17 *Stinson (2007): For a set of positive integers K, let* $\alpha(K) = \gcd\{k - 1 : k \in K\}$ *and* $\beta(K) = \gcd\{k(k-1) : k \in K\}$. *Then the necessary conditions for the existence of a PBD-*(v, K, λ) *are:*

- $\lambda(v-1) \equiv 0 \ (mod\ \alpha(K))$;
- $\lambda v(v-1) \equiv 0 \ (mod\ \beta(K))$.

Example 1.46 *Given* $K = \{3, 4, 6\}$, *we can compute that*

$$\alpha(K) = \gcd\{2, 3, 5\} = 1, \tag{1.32}$$

and

$$\beta(K) = \gcd\{6, 12, 30\} = 6, \tag{1.33}$$

then the necessary conditions for the existence of a PBD-$(v, K, 1)$ *are* $v(v-1) \equiv 0 \ (mod\ 6)$, *which can be simplified to* $v \equiv 0$ or $1 \ (mod\ 3)$.

Theorem 1.18 *De Bruijn and Erdös (1948): For a PBD-*$(v, K, 1)$ *with* b *blocks, where* $b > 1$*, we have* $b \geq v$*. If* $b = v$*, then either the PBD-*$(v, K, 1)$ *has one block of size* $v - 1$ *and the rest of size 2, or else* $K = \{k\}$ *and* $b = v = k^2 - k + 1$*.*

Example 1.47 *The PBD-*$(v, K, 1)$ *with* $v = b = 7$ *only exists as the following two cases:*

$$\begin{aligned} &X_1 = \{1, 2, 3, 4, 5, 6, 7\}; \\ &B_1 = \{12, 13, 14, 15, 16, 17, 234567\}. \end{aligned} \tag{1.34}$$

$$\begin{aligned} &X_2 = \{1, 2, 3, 4, 5, 6, 7\}; \\ &B_2 = \{123, 145, 167, 246, 257, 347, 356\}. \end{aligned} \tag{1.35}$$

Theorem 1.19 *Stinson (2007): Given a resolvable BIBD-*$(v, k, 1)$*, there exists a PBD-*$(v + r, \{k + 1, r\}, 1)$*, where* $r = (v - 1)/(k - 1)$*.*

Example 1.48 *Given the resolvable BIBD-*$(4, 2, 1)$ *in Example 1.27, there exists a PBD-*$(7, 3, 1)$ *with*

$$\begin{aligned} &X = \{1, 2, 3, 4, 5, 6, 7\}; \\ &B = \{123, 145, 167, 246, 257, 347, 356\}. \end{aligned} \tag{1.36}$$

Corollary 1.3 *For any even integer* $v \geq 4$*, given the resolvable BIBD-*$(v, 2, 1)$*, there exists a PBD-*$(2v - 1, \{3, v - 1\}, 1)$*.*

1.6.3.2 Group Divisible Design

Definition 1.20 **Group divisible design:** Suppose $t \geq 2$, $\mu \geq 1$, $K \subseteq \{n \in Z : n \geq 2\}$, and $G \subseteq \{n \in Z : n \geq 1\}$. A group divisible design (GDD-(t, K, G, μ)) is a triple $(X, \mathcal{G}, B)$ that satisfies the following properties:

- X is a set of t elements;
- $\mathcal{G}$ is a partition of X into at least two groups;
- B is a collection of blocks, each of which is a sub-set of elements in X;
- the number of elements contained in each block belongs to the set K;
- the number of elements in each group belongs to the set G;
- every pair of distinct elements is contained in exactly μ common blocks or one group, but not both.

Remark 1.22 For a GDD, it is allowed that each group consists of only one element.

Given a GDD-(t, K, G, λ) with a_i groups of g_i elements, $i = 1, 2, ..., s$, which implies $\sum_{i=1}^{s} a_i g_i = t$, we use the notation $g_1^{a_1} g_2^{a_2} ... g_s^{a_s}$ for the group type of such a GDD.

Example 1.49 *A GDD-*$(10, \{3, 4\}, \{1, 3\}, 1)$ *with group type* $1^1 3^3$.

$$\begin{aligned} &X = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}; \\ &\mathcal{G} = \{\{0, 1, 2\}, \{3, 4, 5\}, \{6, 7, 8\}, \{9\}\}; \\ &B = \{0369, 1479, 2589, 048, 156, 237, 057, 138\}. \end{aligned} \tag{1.37}$$

Theorem 1.20 *Colbourn and Dinitz (2006): Suppose that* $v > k \geq 2$ *and* $u = \frac{v-1}{k-1}$. *Then there exists a BIBD-*$(v, k, 1)$ *if and only if there exists a GDD-*$(v - 1, \{k\}, \{k-1\}, 1)$ *with group type* $(k-1)^u$.

Example 1.50 *Suppose* $v = 7$ *and* $k = 3$, *then we can calculate* $u = 3$. *Since there exists a BIBD-*$(7, 3, 1)$ *in Example 1.32, there exists a GDD-*$(6, \{3\}, \{2\}, 1)$ *with group type* 2^3 *as follows.*

$$\begin{aligned} &X = \{1, 2, 3, 4, 5, 6\}; \\ &\mathcal{G} = \{\{1, 2\}, \{3, 4\}, \{5, 6\}\}; \\ &B = \{135, 146, 236, 245\}. \end{aligned} \tag{1.38}$$

Theorem 1.21 *Zhu (1993): Let* k, m *be integers with* $k \geq 2$ *and* $m \geq 1$. *The necessary conditions for the existence of a GDD-*$(t, \{k\}, \{m\}, \mu)$ *with group type* m^u *are:*

- $u \geq k$;
- $\mu(u-1)m \equiv 0 \ (mod\ k-1)$;
- $\mu u(u-1) \equiv 0 \ (mod\ k(k-1))$.

Example 1.51 *Suppose* $k = 3$ *and* $m = 2$, *then the necessary conditions for the existence of a GDD-*$(t, \{k\}, \{m\}, 1)$ *with group type* m^u *are* $u \geq 3$, $2(u-1) \equiv 0 \ (mod\ 2)$, *and* $u(u-1) \equiv 0 \ (mod\ 6)$, *which can be simplified to* $u \geq 3$ *and* $u \equiv 0$ or $1 \ (mod\ 3)$.

1.6.3.3 Transversal Design

Definition 1.21 **Transversal Design:** Suppose $n \geq 2$, $\mu \geq 1$, and $v \geq 2$. A transversal design (TD-(n, μ, v)) is a triple $(X, \mathcal{G}, B)$ that satisfies the following properties:

- X is a set of nv elements;
- $\mathcal{G}$ is a partition of X into n groups;

- Each group contains exactly v elements;
- B is a collection of μv^2 blocks, each of which is a sub-set of elements in X;
- Each block contains exactly n elements;
- Every pair of distinct elements is contained in exactly μ common blocks or one group, but not both.

Remark 1.23 A TD-(n, μ, v) is a GDD-$(nv, \{n\}, \{v\}, \mu)$ with group type v^n.

Example 1.52 *An example of TD-*$(3, 1, 2)$ *is given as follows:*

$$\begin{aligned} &X = \{1, 2, 3, 4, 5, 6\}; \\ &\mathcal{G} = \{\{1, 2\}, \{3, 4\}, \{5, 6\}\}; \\ &B = \{135, 146, 236, 245\}. \end{aligned} \tag{1.39}$$

Remark 1.24 Given a TD-(n_1, μ, v), then there exists a TD-(n_2, μ, v) for any integer n_2 with $2 \le n_2 \le n_1$ by deleting $(n_2 - n_1)$ groups of elements in the original TD-(n_1, μ, v).

Lemma 1.2 *MacNeish (1922), Stinson (2007): Let an integer* $v > 1$ *with its prime factorization as* $p_1^{e_1} p_2^{e_2} \ldots p_l^{e_l}$*, where the* p_i*s are distinct primes and* $e_i \ge 1$ *for all i. Let* $n = \min\{p_i^{e_i} + 1 : 1 \le i \le l\}$*. Then there exists a TD-*(n, μ, v)*.*

Example 1.53 *If* v *is a prime power, TD-*(n, μ, v) *exists if and only if* $2 \le n \le v + 1$*.*

Definition 1.22 **Resolvable transversal design:** A TD satisfying resolvable property is called a resolvable transversal design. To be specific, blocks in a resolvable transversal design can be partitioned into some groups, such that each group of blocks contains each element exactly once.

Theorem 1.22 *Stinson (2007): Given a prime power* v *and an integer* n *such that* $2 \le n \le v$*, then TD-*(n, μ, v) *is resolvable.*

Example 1.54 *Given* $v = n = 2$*, a resolvable TD-*$(2, 1, 2)$ *is provided as follows:*

$$\begin{aligned} &X = \{1, 2, 3, 4\}; \\ &\mathcal{G} = \{\{1, 2\}, \{3, 4\}\}; \\ &B = \{[13, 24], [14, 23]\}, \end{aligned} \tag{1.40}$$

where the notation [ ] *represents the group of blocks such that each group contains all elements exactly once.*

1.6.4 The Relationships Among Extended Block Designs

Theorem 1.23 *Stinson (2007): If $(X, \mathcal{G}, A)$ is a group divisible design, then (X, B) is a pairwise balanced design with $\lambda = 1$, where*

$$B = A \cup \{g \in \mathcal{G} : |g| \geq 2\}. \tag{1.41}$$

Example 1.55 *Given the GDD-$(10, \{3, 4\}, \{1, 3\}, 1)$ in Example 1.49, the induced PBD-$(10, \{3, 4\}, 1)$ is provided as follows.*

$$\begin{aligned} &X = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}; \\ &B = \{0369, 1479, 2589, 048, 156, 237, 057, 138, 012, 345, 678\}. \end{aligned} \tag{1.42}$$

Theorem 1.24 *Stinson (2007): Suppose that $(X, \mathcal{G}, A)$ is a group divisible design and $\infty \notin X$, define $Y = X \cup \{\infty\}$ and*

$$B = A \cup \{g \cup \{\infty\} : g \in \mathcal{G}\}. \tag{1.43}$$

Then (Y, B) is a PBD with $\lambda = 1$.

Example 1.56 *Given the GDD-$(10, \{3, 4\}, \{1, 3\}, 1)$ in Example 1.49, the induced PBD-$(11, \{2, 3, 4\}, 1)$ is provided as follows.*

$$\begin{aligned} &Y = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, \infty\}; \\ &B = \{0369, 1479, 2589, 048, 156, 237, 057, 138, 012\infty, 345\infty, 678\infty, 9\infty\}. \end{aligned} \tag{1.44}$$

Theorem 1.25 *Stinson (2007): If (X, A) is a PBD with $\lambda = 1$, then $(X, \mathcal{G}, A)$ is a GDD, where*

$$\mathcal{G} = \{\{x\} : x \in X\}. \tag{1.45}$$

Remark 1.25 Any pairwise balanced design is a group divisible design by partitioning elements into singleton groups, i.e., each group consists of exactly one element.

Theorem 1.26 *Stinson (2007): Given an integer $n \geq 2$ and a TD-$(n + 1, 1, v)$, then the following pairwise balanced designs exist:*

- *A PBD-$(nv + 1, \{n, n + 1, v\}, 1)$;*
- *A PBD-$((n + 1)v, \{n + 1, v\}, 1)$;*
- *A PBD-$(nv + u, \{n, n + 1, v, u\}, 1)$ for all u such that $2 \leq u \leq v - 1$.*

Example 1.57 *Suppose $n = 2$ and $v = 3$, then the following $(X, \mathcal{G}, B)$ is an example of TD-$(3, 1, 3)$ with*

$$\begin{aligned} X &= \{1, 2, 3, 4, 5, 6, 7, 8, 9\}; \\ \mathcal{G} &= \{123, 456, 789\}; \\ B &= \{147, 159, 168, 249, 258, 267, 348, 357, 369\}, \end{aligned} \tag{1.46}$$

According to Theorem 1.26, there exist the following pairwise balanced designs:

- *A PBD-$(7, \{2, 3\}, 1)$ with*

$$\begin{aligned} X_1 &= \{1, 2, 3, 4, 5, 6, 7\}; \\ B_1 &= \{123, 456, 147, 15, 16, 24, 25, 267, 34, 357, 36\} \end{aligned} \tag{1.47}$$

- *A PBD-$(9, \{3\}, 1)$ with*

$$\begin{aligned} X_2 &= \{1, 2, 3, 4, 5, 6, 7, 8, 9\}; \\ B_2 &= \{123, 456, 789, 147, 159, 168, 249, 258, 267, 348, 357, 369\} \end{aligned} \tag{1.48}$$

- *A PBD-$(8, \{2, 3\}, 1)$ (i.e., $u = 2$) with*

$$\begin{aligned} X_3 &= \{1, 2, 3, 4, 5, 6, 7, 8\}; \\ B_3 &= \{123, 456, 789, 147, 15, 168, 24, 258, 267, 348, 357, 36\} \end{aligned} \tag{1.49}$$

1.7 Subgraph Options

In the design framework, the family of Ramanujan Graph is selected as the main option of subgraphs. Complete graph(CG) and complete bipartite graph(CBG) are members of this family. Next we will introduce two novel members of such family: block incidence graph(BIG) and transversal design graph(TDG).

1.7.1 Block Incidence Graph

Definition 1.23 A *block incidence graph*, denoted by BIG-(θ, r, k, λ), is defined as a biregular bipartite graph that contains θk nodes with degree r (called *element nodes*) and θr nodes with degree k (called *block nodes*), such that

- all element nodes are contained in one partition of the graph, while the other partition contains all of the blocks nodes;
- each pair of element nodes are adjacent to a number of λ common block nodes;
- the parameters satisfy $\theta, r, \lambda \geq 1$ and $k \geq 2$.

Examples of block incidence graph are shown in the following figure:

The name "block incidence graph" comes from the fact that such a graph can be regarded as the incidence graph of a complete block design (CBD) or a balanced incomplete block design (BIBD) (*cf.* Section 1.6.2). Specifically, given a CBD or a BIBD, its corresponding BIG can be constructed by mapping the elements and blocks in such a design as element nodes and block nodes, respectively. For each element in the CBD (or BIBD), if it is contained in a block, then its corresponding element node and block node are adjacent to each other. For example, a complete bipartite graph (CBG) can be generated by a CBD as a special case of BIG with $\theta = 1$ (e.g., the BIG-(1, 3, 3, 3) in Figure 1.19). It is not hard to obtain the following lemma.

Lemma 1.3 *If $\theta = 1$, any BIG-(θ, r, k, λ) is an incidence graph of a CBD with k elements and r blocks (i.e., a CBG $K_{r,k}$). Otherwise, if $\theta > 1$, then it is an incidence graph of a BIBD-(v, b, r, k, λ) with $v = \theta k$ and $b = \theta r$. Conversely, for any CBD with at least two elements, its incidence graph is a BIG-(θ, r, k, λ) with $\theta = 1$. For any BIBD-(v, b, r, k, λ), its incidence graph is a BIG-(θ, r, k, λ) with $\theta > 1$, $v = \theta k$, and $b = \theta r$.*

Properties of BIGs: To employ BIG as the internal graph of a module, we investigate its expansion property and diameter in the following two lemmas.

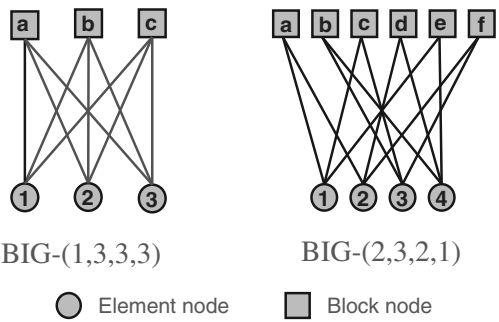

Figure 1.19 Two instances of block incidence graph (denoted by BIG-(θ, r, k, λ)). In the BIG-(1, 3, 3, 3), the number of element nodes θk and the number of block nodes θr are both three. Each pair of element nodes are adjacent to three common block nodes ($\lambda = 3$), including block nodes a, b, and c. As for the BIG-(2, 3, 2, 1), it contains four element nodes ($\theta k = 4$) and six block nodes ($\theta r = 6$). Each pair of element nodes are adjacent to only one common block node ($\lambda = 1$). For example, element nodes 1 and 2 are both adjacent to block node a.

Lemma 1.4 *For a BIG-(θ, r, k, λ), the second largest eigenvalue (SLE) of its adjacency matrix satisfies that*

$$\lambda_2^{eig}(BIG) = \sqrt{\frac{rk(\theta - 1)}{\theta k - 1}},$$

which achieves the tightest lower bound of SLEs for all biregular bipartite graphs given the same number of nodes and the same node degrees in such a BIG.

The proof of Lemma 1.4 is provided in Section A.1.1. The lemma leads to the following corollary whose proof is given in Section A.1.2.

Corollary 1.4 *A BIG-(θ, r, k, λ) is a bipartite Ramanujan graph.*

The diameter of BIG is demonstrated by the following lemma, which is proved in Section A.1.3.

Lemma 1.5 *The diameter of a BIG-(θ, r, k, λ) is at most four. Specifically, the distance between two nodes in a BIG is discussed in the following three cases.*

- *The distance between each pair of element nodes is two.*
- *The distance between an element node and a block node is one or three.*
- *The distance between each pair of block nodes is two or four.*

Based on the properties of CBD and BIBD, it can be shown that the parameters of BIG satisfy the following lemma, which is proved in Section A.1.4.

Lemma 1.6 *For a BIG-(θ, r, k, λ), we have*

$$\lambda = \frac{r(k-1)}{\theta k - 1}. \tag{1.50}$$

Note that BIG does not always exist under any parameter settings (i.e., θ, r, k, λ). When $\theta = 1$, we have $\lambda = r$ according to Lemma 1.6. Such a BIG-$(1, r, k, r)$ is a CBG and universally exists for any integers r, k with $r \geq 1$ and $k \geq 2$. However, when $\theta > 1$, the BIG is induced by some BIBD which does not necessarily exist. By the sufficient conditions for the existence of BIBD, there are some sufficient conditions for the existence of BIG. Such conditions are demonstrated by the following lemma, which is proved in Section A.1.5. Figure 1.20 shows some examples for illustration.

Lemma 1.7 *A number of BIGs can be induced by some well-known BIBDs Colbourn and Dinitz (2006). The sufficient conditions with respect to the existence of the BIGs with $\theta > 1$ and $r = q$ are given as follows.*

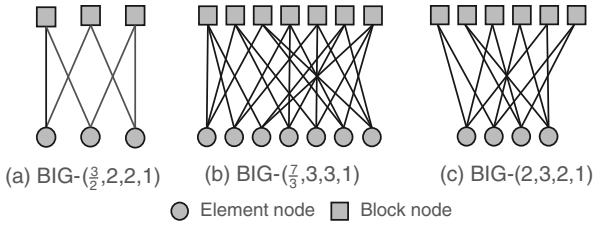

Figure 1.20 Some examples to illustrate the BIGs specified in Lemma 1.7. (a) When $q = 2$, a BIG-$(\frac{3}{2}, 2, 2, 1)$ can be viewed as a special case of a BIG-$(1 + \frac{1}{q}, q, q, q - 1)$, a BIG-$(2 - \frac{1}{q}, q, q, \frac{q}{2})$, a BIG-$(q - 1 + \frac{1}{q}, q, q, 1)$, and a BIG-$(\frac{q+1}{2}, q, 2, 1)$. (b) When $q = 3$, a BIG-$(\frac{7}{3}, 3, 3, 1)$ can be viewed as a special case of a BIG-$(2 + \frac{1}{q}, q, q, \frac{q-1}{2})$ and a BIG-$(q - 1 + \frac{1}{q}, q, q, 1)$. (c) When $q = 3$, a BIG-$(2, 3, 2, 1)$ as a special case of a BIG-$(\frac{q+1}{2}, q, 2, 1)$ and a BIG-$(q - 1, q, q - 1, 1)$.

- *Given an integer q such that $q \geq 2$, there exists a BIG-$(1 + \frac{1}{q}, q, q, q - 1)$.*
- *Given an even number q such that $2 \leq q \leq 212$, a BIG-$(2 - \frac{1}{q}, q, q, \frac{q}{2})$ can be induced by a Hadamard BIBD.*
- *Given an odd number q such that $3 \leq q \leq 211$, a BIG-$(2 + \frac{1}{q}, q, q, \frac{q-1}{2})$ can be induced by a Hadamard BIBD.*
- *Given an integer q such that $q \geq 2$, there exists a BIG-$(\frac{q+1}{2}, q, 2, 1)$.*
- *Given a prime power $q - 1$, a BIG-$(q - 1, q, q - 1, 1)$ can be induced by an affine BIBD.*
- *Given an integer q such that $q = 2$ or $q - 1$ is a prime power, a BIG-$(q - 1 + \frac{1}{q}, q, q, 1)$ can be induced by a projective BIBD.*

1.7.2 Transversal Design Graph

Definition 1.24 A *transversal design graph*, denoted by TDG-(n, μ, v), is defined as a biregular bipartite graph, where nodes in one node partition are called *element nodes* and nodes in the other node partition are called *block nodes*, such that

- there are n groups of v element nodes with degree μv and μv^2 block nodes with degree n;
- two element nodes in distinct groups are adjacent to μ common block nodes;

- two element nodes in the same group have no common adjacent block nodes;
- the parameters satisfy $n, \upsilon \geq 2$ and $\mu \geq 1$.

Examples of block incidence graph are shown in the following Figure 1.21.

The name "transversal design graph" comes from the fact that such a graph can be regarded as the incidence graph of a transversal design (TD). Specifically, given a TD, its corresponding TDG can be constructed by mapping the elements and blocks in such a TD to element nodes and block nodes, respectively. For each element in the TD, if it is contained in a block, its corresponding element node and block node are adjacent to each other. Such a mapping is concluded by the following lemma, whose proof is provided in Section A.1.6.

Lemma 1.8 *Given a TDG-(n, μ, υ), it can be viewed as the incidence graph of a TD-(n, μ, υ). Conversely, for any TD-(n, μ, υ), its incidence graph is a TDG-(n, μ, υ).*

Properties of TDGs:

Lemma 1.9 *For a TDG-(n, μ, υ), the SLE of its adjacency matrix satisfies*

$$\lambda_2^{eig}(\mathrm{TDG}) = \sqrt{\mu\upsilon}. \tag{1.51}$$

The proof of Lemma 1.9 is provided in Section A.1.7. The lemma leads to the following corollary whose proof is given in Section A.1.8.

Corollary 1.5 *A TDG-(n, μ, υ) is a bipartite Ramanujan graph.*

It is not hard to obtain the following lemma:

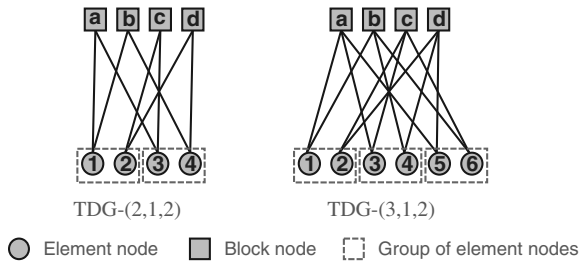

Figure 1.21 Two instances of transversal design graph (denoted by TDG-(n, μ, υ)). In the TDG-(2, 1, 2), the numbers of element nodes $n\upsilon$ and block nodes $\mu\upsilon^2$ are both four. Any two element nodes in distinct groups are adjacent to only one common block node ($\mu = 1$). For example, element nodes 1 and 4 are both adjacent to block node b. As for the TDG-(3, 1, 2), it contains six element nodes ($n\upsilon = 6$) and four block nodes ($\mu\upsilon^2 = 4$). Any two element nodes in distinct groups are adjacent to only one common block node ($\mu = 1$). For example, element nodes 1 and 3 are both adjacent to block node a.

Lemma 1.10 *The diameter of a TDG-(n, μ, v) is at most four. Specifically, the distance between two nodes in a TDG is considered in the following three cases.*

- *For each element node, its distance with any element node in distinct groups is two, while its distance with another element node in the same group is four.*
- *The distance between an element node and a block node is one or three.*
- *For each block node, there exist at most $\mu n(v-1)+\mu-1$ block nodes at a distance of two from such a block node. For the other block nodes, their distances from the block node are all equal to four.*

Moreover, the existence conditions of TDG can be implied from the existence conditions of TD, as given by the following lemma. This lemma is proved in Section A.1.9.

Lemma 1.11 *Regarding the existence of TDG, two sufficient conditions are given as follows, respectively.*

- *If v is a prime power, there exists a TDG-(n, μ, v) for $2 \le n \le v+1$.*
- *If $n = 2$ or $n = 3$, there exists a TDG-(n, μ, v) for any integer $v \ge 2$.*