doi:10.11887/j.issn.1001-2486.25110046 http://journal.nudt.edu.cn

高性能互连网络拓扑研究综述

董德尊*,王梓宇,雷 斐
(国防科技大学 计算机学院,湖南 长沙 410073)

摘 要:高性能互连网络是决定超算与智算系统可扩展性的关键因素之一。拓扑是互连网络可扩展性设计的核心。拓扑设计既要考虑应用与软硬件系统等宏观设计需求,也要考虑路由器芯片端口数、虚拟通道数量、封装密度等多种制约因素。本文较为系统地分析总结了学术和工业界的重要拓扑结构,并对有代表性的新型拓扑进行了详细阐述;分析了自适应路由在高阶网络中的设计难点,对比了典型拓扑的性能和成本,并讨论了选型建议;初步探讨了未来拓扑设计的挑战与发展趋势,包括从智算应用特点出发针对性地设计高性价比的网络拓扑、供电制约的拓扑设计需要拓扑和大楼供电能力的配合、超节点内外网络拓扑的融合与协同设计等。

关键词:高性能互连;拓扑;路由
中图分类号:TP303 **文献标志码**:A **文章编号**:1001-2486(2026)02-266-18

Survey on topology of high-performance interconnection networks

*DONG Dezun**, *WANG Ziyu*, *LEI Fei*
(College of Computer Science and Technology, National University of Defense Technology, Changsha 410073, China)

Abstract: High-performance interconnection networks are among the key factors determining the scalability of supercomputing and intelligent computing systems. Topology serves as the core of scalability-oriented interconnection network design. The design of topology must not only address macro-level requirements from applications and hardware-software systems, but also consider multiple constraints such as router chip port count, number of virtual channels, and packaging density. Significant topological structures from both academia and industry were systematically analyzed and summarized, and a detailed exposition of representative novel topologies was provided. The design challenges of adaptive routing in high-radix networks were examined, the performance and cost of typical topologies were compared, and recommendations for topology selection were discussed. Furthermore, it preliminarily explored future challenges and trends in topology design, including developing cost-effective network topologies tailored to the characteristics of intelligent computing applications, coordinating topology design with building power supply constraints, and integrating and co-designing intra-and inter-supernode network topologies.

Keywords: high-performance interconnect; topology; routing

近年来,高性能计算(high performance computing,HPC)持续快速发展,已迈入后E级时代,并稳步向Z级计算规模演进。随着大语言模型(large language models,LLM)应用等对智算需求的快速增长,人工智能(artificial intelligence,AI)数据中心规模亦高速扩张。xAI、OpenAI、Meta等全球领先企业竞相建设规模超过10万加速卡、功耗达到GW级的AI数据中心集群[1-2]。高性能互连网络是决定超算与智算系统可扩展性的关键因素。互连网络拓扑决定了网络中节点、路由器以及链路的连接方式,是互连网络可扩展性设计的核心。

拓扑的可扩展性设计应同时考虑成本效益、带宽、网络直径等关键特性。胖树(fat-tree)拓扑是当前互连网络事实上的标准方案,其能够在任意流量模式下提供无阻塞的网络带宽,并表现出优异的网络性能。然而,成本始终是互连网络设计的重要制约因素。在面向Z级计算及超10万加速卡互连的过程中,节点规模与计算能力显著提升,随之而来的高带宽需求迫使网络规模进一

收稿日期:2025-11-25
基金项目:国家自然科学基金创新群体资助项目(62421002);国家自然科学基金联合基金重点资助项目(U24B20151);国家重点研发计划资助项目(2022YFB45017202)
*第一作者:董德尊(1980—),男,黑龙江五常人,研究员,博士,博士生导师,E-mail:dong@nudt.edu.cn
引用格式:董德尊, 王梓宇, 雷斐. 高性能互连网络拓扑研究综述[J]. 国防科技大学学报, 2026, 48(2): 266-283.
Citation: DONG D Z, WANG Z Y, LEI F. Survey on topology of high-performance interconnection networks[J]. Journal of National University of Defense Technology, 2026, 48(2): 266-283.

步扩大。对于无阻塞的胖树结构而言,这意味着将消耗数十亿美元的互连成本,因此推动了学术界与工业界对高成本效益拓扑的持续探索。

拓扑设计不仅受制于成本,还受到路由器芯片架构、物理封装密度、路由算法特性、功耗约束以及容错能力等多方面因素的影响。例如,在国产路由器芯片端口数受限的条件下,构建具备高带宽、良好扩展性且成本可控的拓扑是一大挑战;在虚拟通道(virtual channel, VC)数量受限的情况下,实现易部署、精准且拥有丰富路径集的自适应路由算法亦存在难度;面对链路或交换机失效,网络的容错设计同样是重要问题。多种工程性约束叠加,显著增加了拓扑设计的复杂性。

过去数十年间,学术界与工业界已提出数十种具有代表性的拓扑方案,它们在路由器芯片封装形式、端口数量需求、物理布局要求、网络生成方式、光纤类型等方面各有差异。本文对现有的典型拓扑进行了系统分类与综述,随后探讨了拓扑设计中面临的主要挑战,并展望了未来的发展趋势。

1　拓扑发展脉络与分类标准

自 20 世纪 80 年代以来,互连网络领域已提出数十种具有代表性的拓扑结构。在早期的互连系统设计中,路由器芯片通常与计算芯片集成封装。这类系统多采用 Mesh 和 Torus 等规则的多维网格型拓扑结构。随着互连规模的扩大及路由器芯片带宽的迅速提升,Dally 等在 2005 年提出了高阶路由器[3]的概念——在路由器芯片总带宽一定的情况下,将路由器设计成总端口数量较多、单端口带宽较低的方案,并证明:随着网络规模的扩大,采用的高阶路由器能够显著降低网络直径,从而有效减少报文传输延迟。由此促进了拓扑向高阶方向的发展,推动了基于高阶路由器的拓扑(如胖树)的广泛应用,催生了 Flattened Butterfly[4]与 Dragonfly[5]等一系列经典高阶拓扑结构。

此后,学术界又基于组合数学与代数图论提出了多种拓扑结构[6-9]。Singla 等提出了基于随机链路的 Jellyfish[10]拓扑,并证明该拓扑在成本、平均路径长度及网络容错性方面具有优势,随后学术界和工业界对随机拓扑展开了进一步研究[11-12]。Besta 等提出了 Slim Fly[13]拓扑,将能够近似求解度-直径问题的 MMS 图结构[14]引入拓扑构建中。

近年来,以大语言模型为代表的生成式人工智能的迅猛发展,显著推动了 AI 数据中心对互连规模与通信带宽的需求,规模超过 10 万加速卡的数据中心已相继建成与部署。为应对超大规模 AI 数据中心在可扩展性、性能及成本效益方面的挑战,学术界与工业界提出了多种基于应用负载特征优化的网络拓扑设计方案。

本文根据拓扑的主要应用场景,将这些拓扑分为三大类:面向分布式数据中心的拓扑、面向高性能并行计算的拓扑以及面向智能计算的拓扑。下文将分别对这三类拓扑进行系统介绍与分析。在展开讨论之前,先给出部分拓扑参数的定义(见表 1)。

表 1　拓扑参数定义

Tab. 1　Topology parameter setting

参数	描述
D	网络中任何两个路由器间最短路径的最长跳步数,即网络直径
k	路由器端口数
p	终端路由器连接的端点数量
N	网络端点数

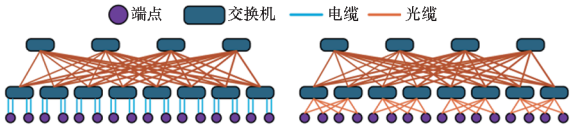
2　面向分布式数据中心的拓扑

拓扑发展历程中提出了多种面向分布式数据中心的网络拓扑,它们通常具有高带宽、可拓展和高容错的特点。胖树拓扑是数据中心网络的典例,其交换机主要分成三类:接入层、汇聚层和核心层。每台交换机的端口数相同,其中,接入层交换机与节点直接连接,汇聚层交换机将接入层交换机向上连接到核心层交换机。每一层网络都具有相同的上下行链路数,这意味着胖树网络具有完全的二分带宽,即对于任何排列的流量模式,至少有一组路径为拓扑中的所有节点都提供完全的可用带宽[15]。但是,胖树拓扑的可拓展性受限于其层数,如式(1)所示,其中 L 表示胖树的层数。

$$N_{\text{fat-tree}(k,L)} = k \times (k/2)^{L-1} \tag{1}$$

多轨胖树是胖树拓扑的一个实用变体。如图 1(a)所示,多轨胖树是指在原有的胖树网络上,端点具有多条独立接入网络的物理链路(轨道)。每条轨道都可以承担一部分流量。实现多轨胖树的常见方法是每个节点配备一个多端口网卡或配备多块网卡。多轨胖树在 HPC 和数据中心领域应用广泛,它的主要优势在于使节点具备更好的负载均衡和冗余容错能力。

一种优化多轨胖树[16-17]的方法是将节点的不同端口接入不同的接入层交换机，这样做可以让节点在一跳内直接到达更多的节点。流量均匀时可以减少数据包平均跳数[18]，还可以降低流量冲突的概率。图 1(b)展示了轨道优化的多轨胖树拓扑。然而，轨道优化的多轨胖树中节点需连接至较远位置的接入层交换机，实际部署中通常采用成本高于电缆的光纤链路，从而增加了光模块的数量与系统总体成本，同时提高了故障风险[19]。



(a) 多轨二级胖树 (b) 轨道优化的多轨二级胖树

(a) Multi-rail two-level fat-tree (b) Rail-optimized multi-rail two-level fat-tree

图 1 多轨胖树和轨道优化的多轨胖树网络示意图

Fig. 1 Illustration of multi-rail fat-tree and optimized multi-rail fat-tree networks

微软公司在 2008 年提出了以服务器为中心的拓扑结构 DCell[20]拓扑。在 DCell 中，交换机之间没有链路连接，服务器节点在网络中既是计算节点也是转发节点。网络被递归构造并可被表示为 DCell(k,L)。图 2 展示了一个 DCell(4, 1)网络，它共有 20 台服务器，并由 5 个仅包含 4 台服务器和 1 个交换机的 DCell(4, 0)网络构成。DCell 网络的可拓展性如式(2)所示。

$$N_{\mathrm{DCell}(k,L)}=\begin{cases}N_{\mathrm{DCell}(k,L-1)}\cdot(N_{\mathrm{DCell}(k,L-1)}+1), L\geqslant 1\\ k, L=0\end{cases} \tag{2}$$

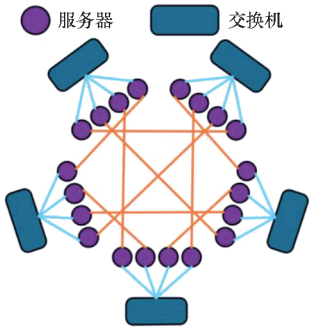


图 2 DCell(4, 1)网络

Fig. 2 DCell (4, 1) network

随着递归层数的增加，DCell 的规模将随交换机端口数呈双指数级快速增长，如一个 DCell(6, 3)网络可容纳超过 300 万台服务器。DCell 网络具备较好的容错性，在面对严重的链路或节点故障时，仍能够实现接近最短路径的路由性能。虽然 DCell 在可扩展性与可靠性方面表现优异，但是其网络布线结构复杂且路径跳数较高，这影响了 DCell 在实际系统中的大规模部署。

在 2009 年，微软公司进一步提出了一种以服务器为中心、可递归构造的 BCube[21]拓扑。与 DCell 相比，BCube 具有更简洁的网络结构、更好的性能和更简单的布线。网络可被表示为 BCube(k,L)，其中 L 表示网络层数和网络直径，网络的可拓展性如式(3)所示。

$$N_{\mathrm{BCube}(k,L)}=k^{L} \tag{3}$$

BCube 的每台服务器配备 L 块网卡，每块网卡连接到不同层的交换机。服务器地址可被编码为一个 L 位的向量。两台服务器之间的路由可被视为对齐两个地址向量中不同位的过程，其中最多有 $b!$ 条最短路径，b 表示地址向量中不同位的数量。图 3 给出了 BCube(4, 2)网络的示意图。BCube 的主要优势在于网络直径小，布线模块化，易于封装。但是，流量在 BCube 路由过程中需要经过 $b-1$ 次服务器内转发，这可能会引入额外的网络延迟，从而造成性能瓶颈。

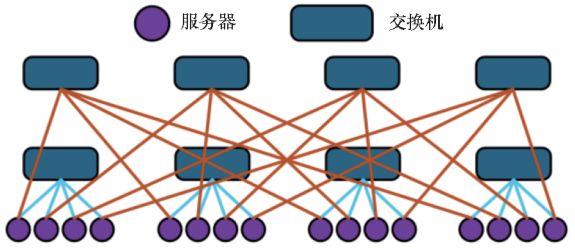


图 3 BCube(4, 2)网络

Fig. 3 BCube(4, 2) network

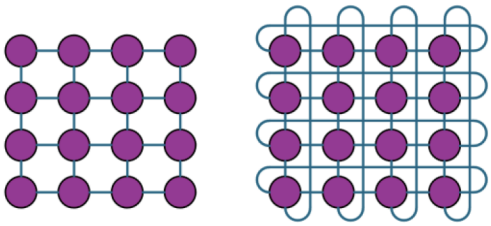
在 2011 年，Singla 等提出基于随机链路的 Jellyfish 拓扑[10]。该架构中每台交换机拥有相同数量的端口，并连接相同比例的端点，交换机之间的链路通过随机方式生成。与胖树网络相比，Jellyfish 在扩展性方面表现更优：在互连相同数量的端点时，可节省 15% ~20% 的交换机和链路；同时，其平均路径长度更短，并具有更强的链路容错能力。此外，Jellyfish 支持简单的增量部署策略：当新增一台交换机时，可随机断开原有网络的若干链路，并将这台新增交换机接入这些断开链路的中继位置即可。

3 面向高性能并行计算的拓扑

3.1 早期低阶拓扑

图 4 为 2D 4×4 Mesh 和 Torus 网络的示意图。二者的区别在于 Torus 网络的每个节点连接

相同数量的链路,且每个维度的边界节点会连接到对面的边界节点,实现“环绕”,这也使得 Torus 具有比 Mesh 更高的二分带宽和更低的直径。



(a) 4×4 Mesh　　(b) 4×4 Torus

图 4　4×4 Mesh 和 Torus 网络示意图

Fig. 4　Illustration of 4×4 Mesh and Torus networks

国际高性能计算排行榜 TOP500 中使用 Mesh 或 Torus 的经典机器包括 Intel 的 Paragon 计算机、IBM 的蓝色基因系列计算机、日本的 K 系列和富岳[22]计算机。Torus 网络还应用于 Google 的张量处理单元[23](tensor processing unit, TPU)系列 AI 数据中心中,利用光交换机来根据应用程序的需求重新配置拓扑。

3.2　高阶路由器概念

高带宽和低延迟始终是互连网络系统的两个重要的性能指标。然而使用低阶路由器所搭建的网络(如高维 Torus)难以在大规模互连网络中保持高的二分带宽且高维度还会导致高直径和高延迟问题。为了简化互连网络的设计并降低网络直径,Kim 在 2005 年提出了高阶路由器的微体系结构[3]。数据包的网络延迟与网络跳步数、路由器芯片带宽、路由器延迟以及数据包长度有关。式(4)给出了数据包网络延迟的表达式。

$$T = T_h + T_s = Ht_r + S/b \tag{4}$$

其中:T_h 表示数据包的零负载网络穿透延迟,它等价于数据包在网络中传输的跳数 H 乘以路由器延迟 t_r;T_s 表示数据包的序列化延迟,等价于 S/b,其中 S 表示数据包的长度,b 表示通道带宽。若使用端口数为 k 的路由器构建规模为 N 的无阻塞网络(如胖树),数据包的跳数至少为 $2\log_k N$[3]。假设路由器带宽为 B 并被所有通道平分,则有 $b = B/(2k)$,将其代入式(4)则有:

$$T = 2t_r \log_k N + 2kS/B \tag{5}$$

令 $\mathrm{d}T/\mathrm{d}k = 0$ 并分离出参数 k 可得令网络延迟最小化的 k 最优值:

$$k\ln^2 k = (Bt_r \ln N)/S \tag{6}$$

其中路由器延迟 t_r 主要由链路层、物理层延迟以及光模块等固定延迟决定,受路由器端口数的影响很小,因此将其视为常数。此外,式(5)中没有考虑数据包在链路上的飞行时间,这是因为数据包在网络中传输的总物理距离与 k 无关。式(6)中 $(Bt_r \ln N)/S$ 被称为纵横比,它决定了最小化网络延迟的路由器端口数。如图 5 所示,当 $k\ln^2 k$ 小于纵横比时,可继续增大路由器端口数 k 以降低网络延迟。当 $k\ln^2 k$ 大于纵横比时,数据包的序列化延迟起主导作用,若继续增大端口数会导致过低的端口带宽所带来的高序列化延迟。

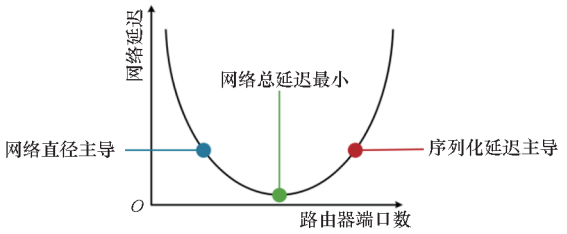


图 5　路由器端口数与网络延迟关系示意图

Fig. 5　Illustration of the relationship between router ports and network latency

此外,发送单个数据包并不是网络通信的常态,式(6)实际应用时还应将 S 替换为最常用发送消息的长度。例如一个具有 51.2 Tbit/s 带宽、400 ns 延迟的路由器芯片和 10 万端点且平均期望消息大小为 16 KB 的网络,则该网络的纵横比为 1 842,对应最优路由器基数为 91。若路由器端口数量满足消息最小化网络延迟,则该端口数的路由器为高阶路由器。学术和工业界后续提出了多种基于高阶路由器的高性能互连网络拓扑。

3.3　层次化的高阶网络

2007 年至 2022 年,超算从 P 级发展至 E 级的过程中,学术界和工业界陆续提出了一系列旨在减少交换机与光缆使用数量的高性能互连网络拓扑。互连网络系统的成本可占总成本的 1/3 以上[13]。相比数据中心,高性能计算系统更加关注使用较低成本来构建低直径高带宽的网络。互连网络系统的成本主要源于交换机和光缆的数量。在高性能互连网络中,链路通常分为全局链路与局部链路两类。其中,全局链路多采用光缆,用于机柜之间的长距离互连;局部链路则通常使用电缆或背板连接。光缆的单价显著高于电缆。例如,截至 2026 年 1 月,一根长度为 30 m、规格为 QSFP 56 200 Gbit/s 的光缆价格约为 1 350 美元,而同等规格、长度为 1.5 m 的电缆价格仅约为 225 美元[24]。因此,拓扑设计的一个重要原则是尽可能减少全局链路和交换机的数量。

Kim 等在 2007 年提出了 Flattened Butterfly[4] 拓扑。Flattened Butterfly 是一个直连网络,其中

网络中所有交换机都连接 $k/(L+1)$ 个端点。Flattened Butterfly 具有多个维度，每个维度的交换机数量都是 $k/(L+1)$ 且它们相互之间全连接。假设 Flattened Butterfly 网络有 L 个维度，则直径也为 L，其拓展性如式(7)所示。

$$N_{\mathrm{FB}(k,L)} = [k/(L+1)]^{L+1} \tag{7}$$

在均匀随机流量模式下，Flattened Butterfly 能够实现和胖树类似的无阻塞网络性能。在对抗性流量模式下，Flattened Butterfly 需要依赖非最小路由来实现负载均衡。由于非最小路由会占用最小路由 2 倍的网络资源，因此，Flattened Butterfly 在对抗性流量模式下的理论最大吞吐率仅为 50%。Flattened Butterfly 部署时可将其中一个维度的交换机和端点封装在机柜内部，机柜内交换机间链路使用背板或短距离电缆连接，而在胖树中相应的链路通常采用光缆连接到中央通信机柜。因此，与胖树相比，Flattened Butterfly 的全局链路数量更少，并可显著节省网络成本[4]。2009 年，Ahn 等提出了 Flattened Butterfly 的拓展拓扑 HyperX[25]。Flattened Butterfly 网络是 HyperX 的子集，其主要区别在于 HyperX 还额外探讨了每个维度的交换机数量、每个交换机所连接的端点数量等多种配置情况。

Dragonfly 拓扑由 Kim 于 2008 年提出[5]，是一种直连层次化拓扑，包含两层全连接结构，并在高性能计算领域被广泛应用。截至 2025 年 11 月，国际高性能计算排行榜 TOP500 中唯四的 E 级计算机[26-28]均采用 Dragonfly 或其变体拓扑。此外，Google 的数据中心网络 Aquila[29] 也基于 Dragonfly 架构。Dragonfly 可由 a、p、h、g 共 4 个参数来唯一确定，并表示为 Dragonfly(a, p, h, g)。网络中共有 g 个组，每个组有 a 个相互全连接的路由器，每个路由器连接 p 个节点，每个路由器有 h 个全局链路连接到其他组的路由器，并形成组间的全连接结构。对于一个最大规模的 Dragonfly 网络，应满足 $a = 2p = 2h$，$g = ah + 1$，$k = a - 1 + p + h$。图 6 给出了一个 Dragonfly(4, 2, 2, 9)网络的示意图。Dragonfly 网络的直径为 3，并具备良好的可拓展性，如式(8)所示。

$$N_{\mathrm{Dragonfly}(k)} = ap(ah+1) \approx k^4/64 + k^2/8 \tag{8}$$

Dragonfly 网络的层次化结构可以简化路由策略并减少用于死锁避免的虚拟通道数量。其最小路由策略定义为：数据包首先经一次组内路由到达具有直连目的组全局链路的组内路由器；随后通过全局链路进入目的组；最终在目的组内再进行一次组内路由到达目的路由器。该过程至多使用一次全局链路和两次局部链路，在由最小路径构成的循环路径中，不存在使用两条全局通道的最小路径，因此最小路由中全局和本地链路分别仅需要一条和两条虚拟通道即可避免死锁。而非最小路由被定义为：选择一个随机的中间路由器，数据包会首先最小路由到中间路由器，随后再次最小路由到目的组。这个过程至多使用两次全局链路和四次局部链路，因此，非最小路由的全局和局部链路分别仅需两条和四条虚拟通道。基于 Dragonfly 网络还提出了一些变体[30-32]，例如将组内结构换成二级胖树可以得到 Dragonfly +[31] 网络，它在保留 Dragonfly 成本效益的同时具有更好的可拓展性，它已被应用于欧洲的 Jupiter[33] 超算上，并在 TOP500（截至 2025 年 11 月）中排名第 4。

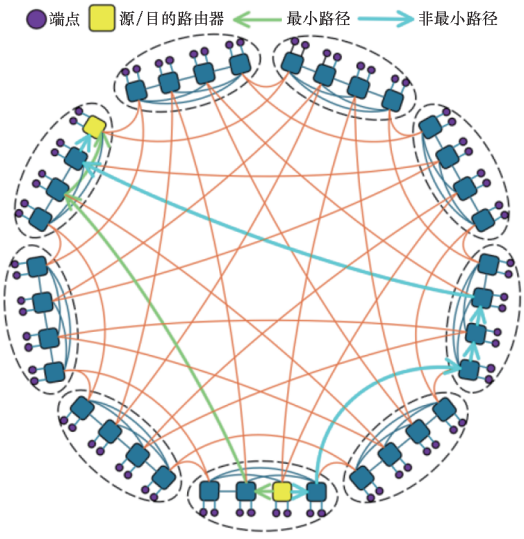


图 6 Dragonfly(4, 2, 2, 9)网络及其最小和非最小路径

Fig. 6 Dragonfly(4, 2, 2, 9) network and the minimal and non-minimal paths

Dragonfly 还在系统封装上具有显著优势，组内的全互连网络可以被封装在一个或多个相邻机柜中，并使用电缆实现组内链路，而昂贵的光缆则用于全局链路连接。平均每端点仅需要 0.5 条全局链路。与 Flattened Butterfly 相比，在实现同等规模时，Dragonfly 网络只需要较少的全局链路数量并可节省 10% ~20% 的网络成本。

此外，设计部署 Dragonfly 的主要挑战之一是设计高效的自适应路由设计。目前学术界已经提出 10 余种 Dragonfly 网络自适应路由算法[34-45]。在 Dragonfly 网络中，路由器间的最短路径数量有限，需要利用非最短路径来实现负载均衡。其中，Dragonfly 的组间全局链路是最易出现拥塞的地方，也是 Dragonfly 路由算法解决的关键。在 5.1 节将更详细地讨论 Dragonfly 及其他高阶网络的

自适应路由设计难点。

国防科技大学“天河”超级计算机研制团队于 2016 年提出了 E 级原型机 Mesh-Tree 网络[46-47],其设计目标是在受限的路由器端口数条件下,构建具备成本效益和高可扩展性的互连网络,以支持 E 级计算。与具有相似可扩展性的四级胖树网络相比,Mesh-Tree 拥有更小的网络平均数据包跳数,并表现出更低的传输延迟。图 7 展示了“天河”E 级原型机的拓扑示意图。该拓扑由多个二维网格排列的组构成,每个组内部采用标准的二级胖树结构。各组的全局端口通过多个全局路由器以均衡方式连接至同列或同行的其他组节点。Mesh-Tree 网络可通过维度顺序路由实现无须虚拟通道的死锁避免,从而降低对路由器虚拟通道数量的配置需求。该网络直径为 6,具备良好的可扩展性,并便于实现模块化部署。

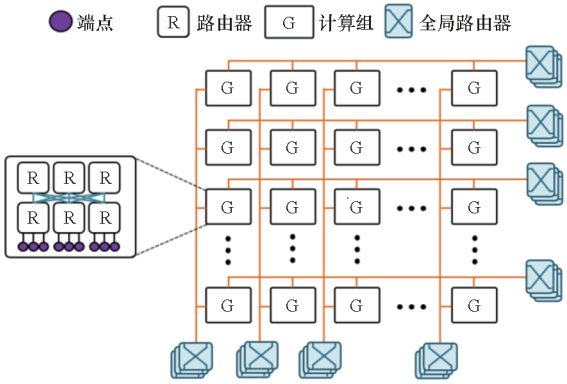


图 7　“天河”E 级原型机 Mesh-Tree 拓扑

Fig. 7　Mesh-Tree topology of Tianhe E-class prototype supercomputer

3.4　基于随机链路的网络

自 2012 年起高性能计算领域出现了一些类似 Jellyfish 的基于随机链路的拓扑[10-11]。与胖树等经典拓扑相比,这类拓扑在相同网络直径下展现出更高的可扩展性、更短的平均路径长度以及更强的网络容错能力。Koibuchi 等于 2012 年提出了分布式环路网络(distributed loop networks, DLN)拓扑[12]。DLN 拓扑中所有交换机都连接相同数量的节点,所有交换机间呈环状排列,随后将交换机剩余的端口以随机链路的形式连接到环的其他交换机上。

Fujiwara 等在 2014 年提出了 Skywalk[11] 拓扑,其研究假设是未来交换机的延迟将可能缩小至 60 ns,在这种情况下,网络延迟主要取决于链路的飞行延迟。Skywalk 也是基于随机链路实现的,但与 Jellyfish 不同,其明确考虑了网络的物理封装。Skywalk 假设机柜呈 2D 网格的形式分布在机房中。交换机间的链路分为 3 种类型,分别是机柜内链路、机柜间直链路(连接两个位于相同行或相同列的机柜)和对角链路(连接不同行且不同列的机柜)。每台交换机连接相同数量的端点,并具有相同数量的 3 种类型的交换机间链路。每一条交换机间链路都是随机连接的。在具体连线中将生成多种连线方式,并选取总链路长度最小的方案。Skywalk 可以实现与 Jellyfish 相似的性能,但从设计时即考虑了模块化的物理封装,避免了极端长度的链路出现。

总体来看,基于随机链路构建拓扑提供了一系列的优点,如更短的平均路径长度、更好的容错、更方便的网络增量部署等,然而这些拓扑的实际部署仍面临挑战。在成本方面,这类拓扑的重要假设是低成本的铜缆长度可达 10 m,但是,随着链路传输速率的提升,铜缆可支持的最长通信距离下降。例如,在主流的 400 Gbit/s 链路速率下,有源铜缆的最大长度仅为 7 m;在 800 Gbit/s 速率下铜缆最大长度仅为 5 m。若要保证更长距离的通信则需要采用成本更高的光缆。成本估算模型也随着链路速率的提升而改变。另外,这类拓扑在路由设计上存在局限性,基于随机链路生成的结构不能像 Dragonfly 那样借助层次化的网络结构来保证路径多样性的同时减少虚拟通道的使用。因此,在随机拓扑中的无死锁路由是通过无环算法静态计算生成的,其路径多样性受限于路由表条目数。

3.5　基于图论的高阶网络

为了更加有效折中成本效益、网络性能和可扩展性,一类基于数学图论的高摩尔界效率网络拓扑[6-9, 13, 48]被提出。摩尔界是指给定网络直径 D 和路由器端口数 k 下,网络所能支持的端点数量的上界。

式(9)给出了摩尔界的表达式,N_r 表示网络中的路由器数量,式(10)中的 k'表示路由器中连接到其他路由器的端口数。对 k'求导并令导数为 0 可得到式(11)的极大值。在给定路由器端口数和网络直径下,分别计算出拓扑的最大网络规模和式(11)的值,将前者除以后者即得到拓扑的摩尔界效率。更高的摩尔界效率意味着互连同等数量的节点时,可以使用更少的交换机和链路,这意味着更高的成本效益。

$$N_r \leqslant 1 + k' \sum_{i=0}^{D-1} (k' - 1)^i \tag{9}$$

$$k' = k - p \tag{10}$$

$$N = p \times N_r \leqslant (k - k') \times \left[1 + k' \sum_{i=0}^{D-1} (k' - 1)^i\right] \tag{11}$$

Besta 等在 2014 年提出了基于 MMS 图构建的 Slim Fly 拓扑[13]。MMS 图由 McKay、Miller 和 Širáň 提出[14]。Slim Fly 的网络直径为 2，具有良好的可拓展性并实现了约 90% 的摩尔界效率。与 Dragonfly 网络相比，Slim Fly 在相似性能下能够节省约 25% 的成本。然而使用 Slim Fly 构建 10 万端点以上的大规模互连时，需要路由器至少具备 88 个端口，这对路由器芯片设计带来了挑战。

为了在路由器端口数、网络直径、可拓展性等方面实现更优的平衡，国防科技大学高性能网络与体系结构（High Performance Networking and Architecture，HiNA）课题组在 2016 年提出了 Galaxyfly[6] 拓扑。Galaxyfly 使用了 Galaxy 图作为顶层结构，Galaxy 图中的节点可以由类似 Dragonfly 中的虚拟路由器组成，其中每个虚拟路由器代表一个全连接网络。Galaxyfly 的最终直径与 Galaxy 图和虚拟路由器的参数有关，可生成直径为 2 ~ 5 的网络。Galaxyfly 具有较好的可拓展性，使用 48 端口路由器，可支持 1 000 ~ 1 000 万端点的网络规模。Galaxyfly 还实现了接近 Slim Fly 的成本效益，并显著降低了对路由器端口数的要求。

随着高性能互连网络规模不断扩大，光纤的数量也随之大幅增长，这对光缆的物理封装和维护性提出了挑战。多芯光纤作为一种新型器件可以等价于一捆单芯光纤，有效降低网络布线复杂性和节省封装成本。因此，国防科技大学 HiNA 课题组在 2020 年提出了 Bundlefly[7] 拓扑。Bundlefly 由 MMS 图和 Paley 图通过多星积的方式构建，能够很好地支持模块化封装，且模块间的多链路能够有效利用多芯光纤来布线。

Lakhotia 等在 2022 年提出了基于Erdős-Rényi 图的直径为 2 的 PolarFly[8] 拓扑。与 Slim Fly 相比，PolarFly 实现了约 96% 的极致摩尔界效率且具有更简单的封装和增量部署策略。Lakhotia 等在 2024 年进一步提出了直径为 3、具有极致摩尔界效率的 PolarStar[9] 网络，并同样支持模块化封装和多芯光纤布线。与传统的对称拓扑（如 Dragonfly 和胖树）相比，基于数学图论方式构建的拓扑具有更高的摩尔界效率，然而其网络结构复杂，不易于维护和封装，且在路由算法设计上难以优化。

4 面向智能计算的拓扑

不同于数据中心或高性能计算领域的拓扑，面向智能计算的新型拓扑通常从目标应用的通信特点出发，针对性地定制结构，以获得最佳的性能与成本比。

Hoefler 等于 2024 年提出了面向深度学习任务的 HammingMesh[49] 拓扑。其核心设计理念在于保证局部带宽，不需要无阻塞胖树结构的高二分带宽，因为深度学习训练过程中大量的环形通信模式对局部带宽的需求显著高于全局带宽。在 HammingMesh 中，多个计算板之间以二维网格的形式排列，其中同属于 X 或 Y 维度的计算板之间使用交换机实现全连接。每个计算板是多个加速器以二维 Mesh 的形式封装在廉价的印制电路板（printed circuit board，PCB）上。图 8 展示了一个共有 $x \times y$ 块计算板的 H $\times$2Mesh 网络（H $\times a$Mesh 表示每块板有 $a \times a$ 个加速器）。HammingMesh 以廉价的 PCB 实现了网络的高局部带宽，并以适度削减全局带宽的方式显著节约了网络成本。对于具有环形通信特征的深度学习应用，HammingMesh 在成本效益方面优于其他常见拓扑结构。

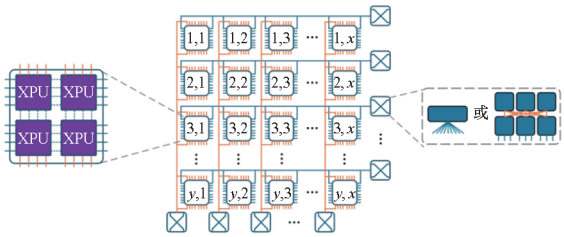


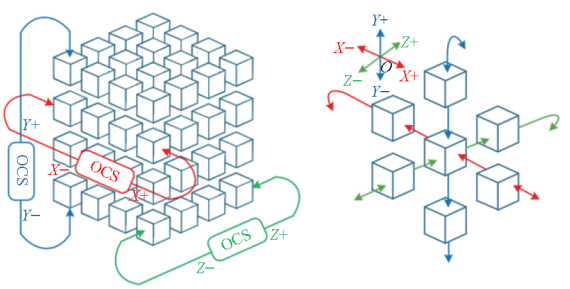
图 8 $x \times y$ H $\times$2Mesh 网络

Fig. 8 $x \times y$ H $\times$2Mesh network

光交换机（optical circuit switch，OCS）概念的提出已逾 50 年，不同于传统的电分组交换机（electronic packet switches，EPS），OCS 将光源从输入引导到输出，数据交换过程完全在光域进行，无须电子分组处理。在早期，光学器件的成本较高、稳定性不足且链路速率较低，EPS 已能满足互连需求。然而随着互连规模的扩大，链路速率持续增长，EPS 频繁的光电转换带来的成本、功耗和可拓展性问题日益突出，OCS 被重新考虑应用于大规模高性能互连系统中[23]。与 EPS 相比，当前 OCS 可达数百个端口[30, 50]，省去了光电转换过程，不仅可节约一半的光模块数量，还显著降低了延迟与能耗。同时，OCS 与比特率、协议、线路编码和调制格式无关，允许光模块与交换机独立演进。微机电系统是当前实现 OCS 的主流技术路

线之一。其基本原理是在 OCS 内布置大量可控的微型反射镜,通过调节反射镜角度,将入射光束引导至不同输出端口以实现光路切换。然而光路切换需要毫秒级的延迟[51],而端到端网络延迟通常是微秒级,二者相差了近千倍,严重局限了 OCS 的使用。

针对机器学习系统在互连成本、性能和可拓展性方面的挑战,谷歌在 2023 年发布了面向机器学习的基于高维 Torus 拓扑和 OCS 互连的 TPU v4 超级计算机[23]系统。其不仅保持了 Torus 网络局部通信带宽优势,还通过 OCS 重构拓扑以优化集合通信的网络需求。TPU v4 共有 4 096 个 TPU 芯片,由 64 个包含 64 个 TPU 芯片的 4 × 4 × 4 Cube 构成。图 9 展示了 TPU Cube 和 TPU 单元的示意图。每个 TPU 芯片可被理解为一个小立方体,立方体的 6 个面代表着 6 条双向链路,一个 Cube 由 64 个 TPU 组成并有总计 384 条双向链路,其中在 Cube 最外表面的 96 条(每个面 16 条,总计 96 条)为全局链路,由 OCS 连接到远程的其他 Cube 上。每个 Cube 的 X、Y 和 Z 轴表面相同位置的立方体通过相同的 OCS 连接,网络共有 48 个 OCS,每个 OCS 具有 128 个输入和输出端口。对于当前最新的 TPU v7 系统,其 Cube 内的连接是相似的,不同之处在于 OCS 的输入输出端口数提升到了 288。值得注意的是,OCS 进行的是输入和输出端口的匹配,这意味着位于 $X+/Y+/Z+$ 表面的 TPU 只能通过 OCS 连接到位于 $X-/Y-/Z-$ 表面的 TPU,在同一个 Cube 的 $X+/Y+/Z+$ 表面的不同 TPU 或在不同 Cube 中相同 $X+/Y+/Z+$ 位置的 TPU 之间不能通过 OCS 进行通信[50]。具有 4 096 个芯片的最大尺度的 TPU v4系统与同时期相似尺寸的英伟达 A100 系统相比,功耗降低了23.0% ~47.3%,在 BERT 模型训练中性能提升了 1.15 倍,在 ResNet 模型训练中性能提升了 1.67 倍。



(a) 4 × 4 × 4 Cube 示意图　(b) TPU 单元连接图

(a) 4 × 4 × 4 Cube schematic　(b) TPU unit interconnection diagram

图 9　谷歌 TPU Cube 和单元示意图

Fig. 9　Illustration of the Google TPU Cube and unit

Wang 等在 2024 年提出了 Rail-Only[52]拓扑,该结构面向 LLM 训练的互连需求,并被 DeepSeek 运用于构建千卡规模的计算集群[53-54]。Rail-Only 的核心设计理念在于 LLM 训练会产生稀疏的通信模式,即在网卡互连层面,任意两块图形处理单元(graphics processing unit, GPU)之间并不必须具备完整的二分带宽。图 10 给出了 Rail-Only 网络的示意图。Rail-Only 可视为在胖树拓扑中去除最顶层交换机的变体,其在超节点内部定义了高带宽域,在该域内,任意两块 GPU 之间的带宽显著高于跨超节点的 GPU 之间的带宽。Rail-Only 可通过在高带宽域内转发流量来实现 LLM 训练产生的全对全通信。与多轨胖树网络相比,其性能仅下降了 4.1% 却节省了 38% 的网络成本。Rail-Only 也可被理解为一种多平面胖树,其中不同节点内相同位置的 GPU 属于同一平面[53-54]。使用单端口网卡构建 Rail-Only 网络时,由于平面不连通,跨平面的通信需要借助节点内的高带宽域转发流量。此过程涉及节点内外不同网络协议的转换和内存拷贝开销,会引入额外延迟。DeepSeek[54]提出了一种可将网卡的单个逻辑端口划分为多个物理端口的理想 Rail-Only 网络,其中网卡的不同物理端口属于不同平面,这使得单网卡连通了所有平面,跨平面的流量不再需要经节点内网络转发。这种网络具有更好的性能、容错和负载均衡,不过该设计要求网卡具备接收乱序数据包的支持[54]。

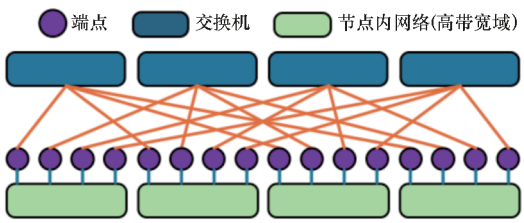


图 10　两层 Rail-Only 网络

Fig. 10　Two-layer Rail-Only network

阿里巴巴在 2024 年提出了专为 LLM 训练的 HPN7.0[55]网络,如图 11 所示。HPN7.0 由多个组组成,每个组由 15 个段组成。每个段由 128 个节点构成,节点总计配备1 024块 GPU;每个节点拥有 8 块 GPU,并配置 9 块双端口网卡,其中 1 块网卡用于连接前端网络。HPN7.0 使用了轨道优化的技术,双端口网卡的两个端口连接到不同平面的接入层交换机上。一个段的 128 个节点通过 16 个接入层交换机连接,并向上提供 960 个 400 Gbit/s端口;15 个段互连成一个组,其中组内

的 120 台汇聚层交换机分别具有 120 个下行和 8 个上行端口，速率均为400 Gbit/s。多个组之间通过核心层交换机连接。组内的网络具有轻微的 1.067∶1 的超额订阅，而组间的网络具有 15∶1 的超额订阅。HPN7.0 的设计动机之一是针对 LLM 训练中周期性出现的“大象”流量。这种长流会使得传统数据中心胖树网络常用的等成本多路径(equal-cost multi-path, ECMP)负载均衡方案容易出现哈希极化效应，从而显著影响模型训练效率。HPN7.0 通过双端口构建的双平面特性能够有效缓解该问题。具体而言，每个网卡的不同端口属于不同的路由平面，当流量从某一端口发送时，网络会沿确定的路径传输，并最终抵达目标网卡相同位置处的端口，从而确保路径的一致性与隔离性。实验表明双平面的设计使柜顶交换机不同入口处的流量变得更均匀且下游端口的队列长度减少了91.8%。进一步的消融研究显示，双平面设计在跨段交通中可提升高达 71.6% 的性能。

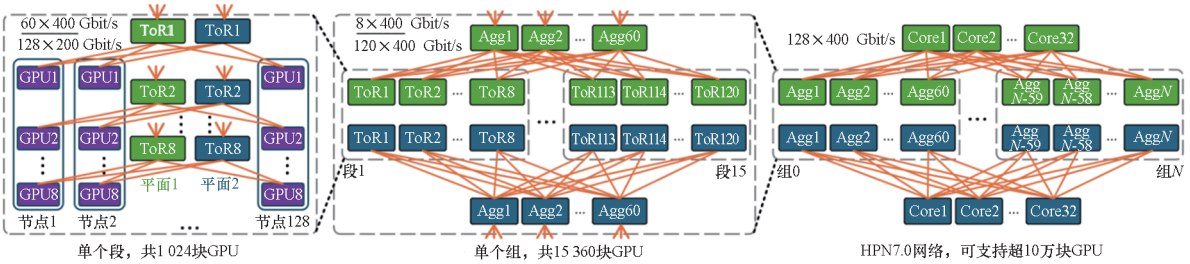


图 11 阿里巴巴 HPN7.0 网络

Fig. 11 Alibaba HPN7.0 network

清华大学网络体系结构、系统和协议(Network Architecture, System and Protocol, NASP)实验室最近提出了 ZCube[56] 网络，它与多轨三层胖树相比具有相似的性能却节省了26% ~ 46% 的成本。ZCube 与 BCube 类似，每台交换机都连接到端点，但在 ZCube 中相邻两层的交换机之间也相互连接。网络可用 ZCube(k,L,n)表示。其中 k、L、n 分别表示交换机端口数、网络层数和网卡端口数，$n \geqslant L$。图 12 给出了 ZCube(8, 2, 2)网络的示意图，其直径为 $L-1$，可拓展性如式(12)所示。

$$N_{\mathrm{ZCube}(k,L,n)} = \begin{cases} (nk/2)^L, L=2 \\ (nk/3)^L, L \geqslant 3 \end{cases} \tag{12}$$

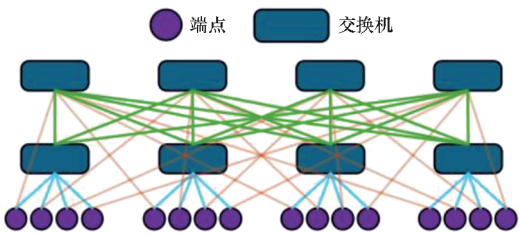


图 12 ZCube(8, 2, 2)网络

Fig. 12 ZCube(8, 2, 2) network

当 ZCube 网络仅包含 2 层结构时，交换机的一半端口用于连接端点，另一半端口连接到相邻层交换机上。然而当网络有 3 层及以上时，非边缘层的交换机将同时连接到端点以及上下两层交换机上，因此可拓展性的表达式与两层 ZCube 网络不同。两层 ZCube 网络具有显著的成本效益，它将每个交换机和网卡的高速端口拆分为多个低速端口，从而使得交换机的可用端口数翻倍，进而能够互连更多端点。例如，基于无阻塞胖树拓扑使用 128 × 400 Gbit/s的交换机构建具有 16 384 块 GPU 的中型数据中心需要 3 层交换网络，而 ZCube 仅需 2 层。ZCube 与 BCube、Rail-Only、HPN 等拓扑相比的另一优势是它无须利用节点内网络中转流量，且直径更小。此外，不同于传统拓扑设计方法，ZCube 是由 NASP 实验室开发的一款自动化网络架构优化设计软件——ATOP 根据 10 多种约束条件搜索得来的。除 ZCube 以外的其他场景，ATOP 还能根据不同的优化目标和限制条件给出组网架构的建议。

在“天河”E 级原型机网络中，不同行和列所属组之间的路由需要维度转置，这导致路径跳步数较高。其实，如何优化这类结构是业界和学术界一直关注的问题。如图 13 所示，一种极致的低直径拓扑设计是使用数量尽可能少的全局路由器，使网络中任何两个组之间的路由不需要维度转置，仅通过至少一个全局路由器芯片一跳可达。

富士通在 2014 年提出了多层全网格[57-58](multi-layer full-mesh, MLFM)，其给出了一种近似最少化全局路由器数量的连接方案，使得任何两个本地路由器可通过一个全局路由器一跳可达[58]。然而，MLFM 是直径为 2 的网络，可拓展性受限，难以满足未来超大规模互连网络的需求。为此，国防科技大学“天河”超级计算机研制团队的董德尊等在 2025 年提出了面向大规模高性能

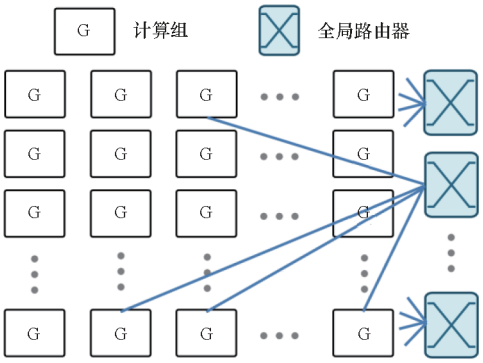


图 13　低直径组间互连示意图

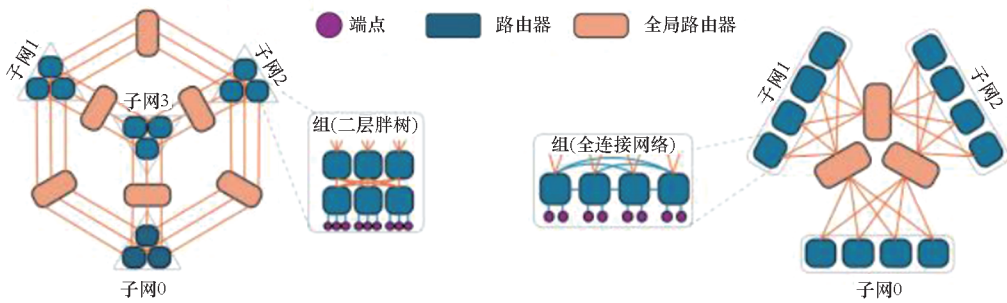
Fig. 13　Illustration of low diameter inter-group interconnection

计算和智能计算系统的 Zettafly 拓扑[59]。图 14给出了 Zettafly-4 和 Zettafly-3 网络的总体以及子网 0 的示意图。Zettafly 网络由 a、p、h、s 共 4 个参数唯一确定,并表示为 Zettafly(a,p,h,s)。Zettafly 网络中每个组有 a 个路由器。网络中每个与端点有连接的路由器连接 p 个端点,每个与全局路由器有连接的路由器有 h 条全局链路,整个网络共有 s 个子网。Zettafly 网络的本地组可以是全连接网络或二级胖树或任何低直径无阻塞网络。当选择全连接网络(或二级胖树)作为组时,Zettafly 保证直径为 3(或 4),简称为 Zettafly-3(或 Zettafly-4)。此外,MLFM 网络可被视为 Zettafly 的特例,其中每个组仅包含单个路由器。

如图 14(c)所示,Zettafly-4 的多个组构成了一个无阻塞子网,其类似一个一半根交换机都悬空的三级胖树。整个网络由多个无阻塞子网构成,每个子网之间通过全局路由器全连接,每个全局路由器的端口被两个子网平分。如图 14(a)和(c)所示,网络中共有 4 个子网,每个子网连接到 9 个全局路由器,其中每 3 个一组连接到其余 3 个子网。Zettafly 网络的可拓展性如式(13)和式(14)所示,使用 64 端口路由器,Zettafly-3 和 Zettafly-4 分别支持 27.8 万和 108 万个端点。在支持中小作业的网络隔离上,Zettafly 具有显著优势。中小作业产生的流量仅使用到一小部分的网络,若为全网络都提供完全的二分带宽是不必要的。但是,现存的拓扑如 Dragonfly、HyperX 等具有完全二分带宽的无阻塞区域又太小,难以对多并发的中小作业做到有效的分区隔离。而 Zettafly 网络由多个大型的无阻塞子网构成,这使得中小作业能够很容易地被隔离在单个无阻塞子网内。Zettafly 还配备了高效的自适应路由算法,经自研全栈仿真平台验证[60-61],Zettafly 在并发多作业的混合流量模式下实现了显著的成本效益。

$$N_{\text{Zettafly-3}(k)} = k^4/64 + k^3/16 \tag{13}$$

$$N_{\text{Zettafly-4}(k)} = k^4/16 + k^3/8 \tag{14}$$

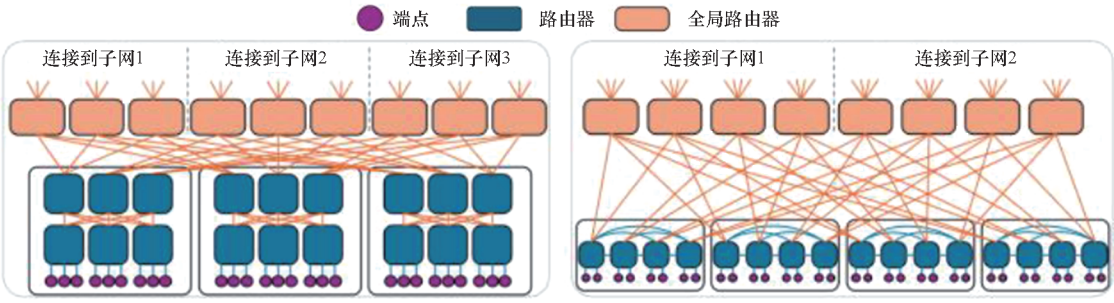


(a) Zettafly-4 (6,3,3,4)总体示意图

(a) Zettafly-4 (6,3,3,4) overall schematic

(b) Zettafly-3 (4,2,2,3)总体示意图

(b) Zettafly-3 (4,2,2,3) overall schematic



(c) Zettafly-4 (6,3,3,4)子网 0 示意图

(c) Zettafly-4 (6,3,3,4) sub-network 0 schematic

(d) Zettafly-3 (4,2,2,3)子网 0 示意图

(d) Zettafly-3 (4,2,2,3) sub-network 0 schematic

图 14　Zettafly-3 (4,2,2,3)和 Zettafly-4 (6,3,3,4)网络

Fig. 14　Zettafly-3 (4,2,2,3) and Zettafly-4 (6,3,3,4) networks

华为在 2025 年提出了专为 LLM 训练而设计的 UB-Mesh[62] 拓扑。不同于传统数据中心网络，UB-Mesh 采用了一种分层局部化的 n 维全网状网络拓扑结构。这种设计充分利用了 LLM 训练的数据局部性，优先采用短距离、直接的互连方式，最小化数据传输跳数和减少交换机数量。图 15 展示了一个具有 16 个机架、1 024 块 GPU 的 UB-Mesh 组示意图。UB-Mesh 中的所有可互连设备都基于统一总线（unified bus，UB）互连，这消除了传统的 AI 数据中心中 PCIe、NVLink、IB/RoCE 等多种互连协议并存所带来的协议转换开销。UB-Mesh 中的 NPU 和 CPU 都集成了 UB-IO 控制模块，其分别具有 72 和 32 条 UB 通道。不同维度间的带宽分配可通过 UB-IO 中预留的通道数量来灵活调整。UB-IO 还具备路由功能，可视为一种低阶路由器。在构建小规模网络时，UB-Mesh 网络可只有前 3 维并可通过 UB-IO 构建直连网络。当互连更大网络时，机架内将配有多个具有 72 条 UB 通道的低阶路由器，这些低阶路由器连接到机架内的 NPU 并使 4 ×4 个机架互连形成一个 4 维的 UB-Mesh-Pod。组内的链路均使用低成本的电缆和低阶路由器互连，而组间互连（图 15 中未绘出）使用光缆和具有 512 条 UB 通道的高阶路由器连接。UB-Mesh 还使用全路径路由（all-path-routing，APR）算法，允许流量在网络中绕路以获得更高带宽。在 MOE-10T、LLAMA2-70B 等多个 LLM 训练任务上，对 8 K 至 10 M 序列长度的性能评估结果表明，与无阻塞胖树相比，UB-Mesh 可达到胖树网络 93% 的训练吞吐率并实现 2.04 倍的成本效益。

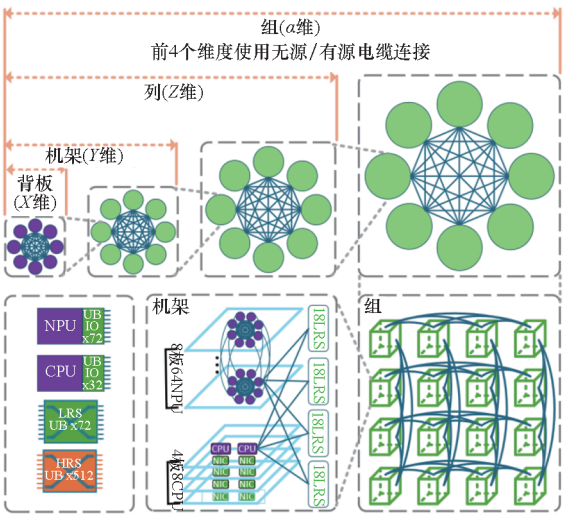


图 15 UB-Mesh 组示意图

Fig. 15 Illustration of UB-Mesh pod

5 分析、对比和总结

数十年来学术界和工业界提出了 30 余种代表性拓扑结构，其中仅有 Mesh/Torus、胖树及其变体以及 Flattened Butterfly、Dragonfly/Dragonfly + 网络被大规模实际部署过，其余拓扑结构仅停留在学术研究或小型原型机层面。本节将进一步分析和总结各类拓扑的优缺点，以供系统设计者们参考并根据需求选型拓扑网络。

5.1 高阶网络与自适应路由的协同设计

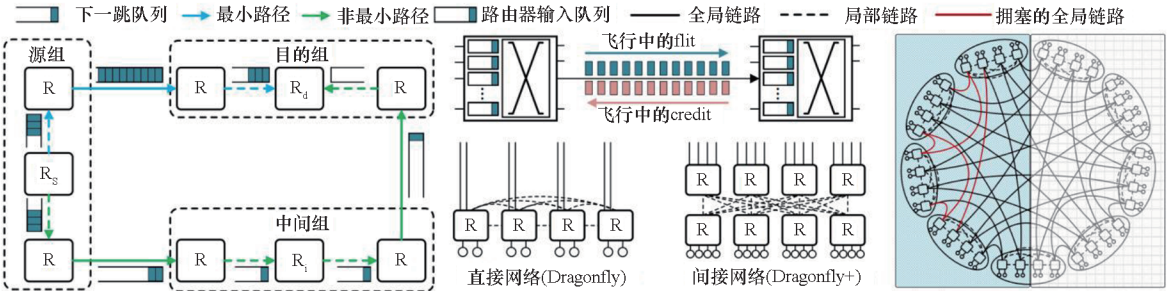
拓扑决定了网络的性能上限，而路由算法则对能否逼近甚至实现这一性能上限起着关键作用。路由算法的性能表现与设计复杂度实际上是制约众多高阶网络在大规模实际部署的重要因素。高阶网络（如 Dragonfly/ Dragonfly + 、Slim Fly、Flattened Butterfly 等）与胖树网络相比具有更优异的成本效益，但这种优势是以牺牲最短路径的多样性为代价换取的。因此，高阶网络需要自适应路由算法才能实现在对抗性流量模式下的负载均衡。然而，在高阶网络中设计简单、高效且实用的自适应路由算法面临诸多挑战。

通用全局自适应负载均衡（universal globally adaptive load-balancing，UGAL）算法[63] 是包括 Dragonfly、Flattened Butterfly、Slim Fly 在内的多种高阶网络的默认自适应路由算法，并已被实际部署在大型互连系统上[64]。UGAL 根据路径下一跳队列长度乘以路径跳数评估候选路径的拥塞情况并据此做出自适应路由决策，具体地，当不等式（15）满足时 UGAL 将最小路由。

$$q_{\mathrm{m}} \times H_{\mathrm{m}} \leqslant q_{\mathrm{nm}} \times H_{\mathrm{nm}} + b \tag{15}$$

其中，q_{m}（q_{nm}）表示最小（非最小）路径根据端口已使用的信用估计的第一跳队列长度；H_{m}（H_{nm}）表示最小（非最小）路径的跳数；b 表示静态偏置或阈值，其为正数将使得 UGAL 偏向于最小路由，为负数则偏向于非最小路由。

原始的 UGAL 性能较差，学术界已提出 10 余种针对 Dragonfly 网络改进 UGAL 的自适应路由算法[34-45]。图 16 总结了 UGAL 在 Dragonfly 网络中存在的三点问题：①当最小路径的全局链路出现拥塞但并未反压到源路由器时，源路由器仍将继续最小路由流量并加剧全局链路拥塞；②长的链路延迟使得大量数据包和信用在飞行中并造成了幻影的拥塞；③网络中非对称流量模式（UR_half），如网络被两个应用所共享，其中右侧为一个几乎不产生流量的计算密集型应用，而左侧是一个通信密集型应用并在内部产生均匀随机流



(a) 间接路由问题　(b) 幻影拥塞问题　(c) 非对称流量问题

(a) Indirect routing problem　(b) Phantom congestion problem　(c) Asymmetric traffic problem

图 16　UGAL 在 Dragonfly 网络的三个问题

Fig. 16　The three problems of UGAL in Dragonfly networks

量,从而形成了一个部分全局链路被超额订阅的 Dragonfly 子网。“下一跳队列”表示路由器根据本地剩余信用估计下游路由器队列长度,而“路由器输入队列”则表示下游路由器输入队列缓冲区的真实长度。全局和局部链路分别用实线和虚线表示。值得注意的是,这些问题不仅在 Dragonfly 网络中存在,也存在于 Dragonfly +、Flattened Butterfly、Slim Fly 等多种高阶网络中。

1) 间接路由问题:图 16 (a)展示了 UGAL 在 Dragonfly 中的间接路由问题。该问题在 Dragonfly 的初始文献[5]中就已发现,并被后续多种自适应路由算法重点研究[35, 38, 40 - 41, 44, 65]。Dragonfly 中最小路径的全局通道常位于组内其他路由器上,因此源路由器只能通过局部通道的背压感知全局通道的拥塞,在此过程中大量数据包都将被最小路由从而使得网络吞吐率下降和延迟增加。间接自适应路由算法是指自适应路由决策使用源路由器无法直接获得的信息,能够有效解决该问题。

2) 幻影拥塞问题:UGAL 使用本地剩余信用来估计下游路由器输入队列的长度,然而这种估计只在通道中不存在飞行中的数据和信用时才是准确的。如图 16 (b) 所示,假设网络处于没有拥塞的良性流通的情况,下游路由器每接收一个数据包都会将其立即转发,因此其输入队列空闲。然而由于长的通道延迟,存在大量飞行中的数据包和信用,这使得上游路由器错误地估计出下游路由器具有较高的输入队列。通道中飞行的数据和信用数量的上限取决于通道延迟,而局部通道的通道延迟显著小于全局通道,最终幻影拥塞会使得 UGAL 偏向于第一跳使用局部通道的路径[35 - 36, 59],并由此做出不精确的路由决策。这种情况在层次化的直接拓扑如 Dragonfly 中不可避免。因为在层次化的直接拓扑中,每个路由器都连接了相同数量的端点和局部/全局链路,则一定存在最小和非最小路径的第一跳使用不同类型通道的情况,而在间接拓扑中情况有所缓解。一种滑动窗口技术[35]被提出来解决这种幻影拥塞,通过在最近一个通道往返延迟窗口内统计发送至该通道的 flit 数量,使路由器能够区分通道的实际拥塞状态与存在大量飞行数据包的状态,从而有效缓解幻影拥塞问题。

3)非对称流量问题:在对自适应路由算法的评估中通常仅使用对称负载,其中网络的每个端点具有相同的注入率和流量模式。然而,在真实系统中,网络承载着通信特性各异的多个应用程序,导致流量负载呈现强烈的不对称性。图 16 (c)展示了 Dragonfly 网络中一种非对称流量模式的示例,其中网络左侧运行着通信密集型应用(记为 app1),而右侧运行着计算密集型应用(记为 app2)。对于 app1 内的流量,存在一些穿过网络二分切割,经过 app2 的区域最后返回 app1 的非最小路径,这些非最小路径在 app2 内的跳数应该具有极低的队列长度。然而 UGAL 根据第一跳的队列长度来近似路径所有跳的队列长度,当自适应路由 app1 内的流量时,UGAL 会高估这些非最小路径的拥塞情况,导致过多的流量被最小路由。

智能路由算法[38, 66]利用目的路由器将端到端网络延迟通过数据包或信用捎带给源路由器,再通过梯度下降或强化学习的方式处理这些网络延迟信息并使得源路由器了解网络的整体流量情况以解决非对称流量问题。

表 2 展示了 Dragonfly 网络多种自适应路由算法的性能和开销比较[35,38,40 - 41,44,63,65 - 67]。在“空间复杂度”一栏中,k 表示路由器端口数,T 表示滑动窗口长度。若算法不使用浮点计算单元,则被视为低逻辑开销(√),否则归为高逻辑开销

表2 Dragonfly网络自适应路由算法性能与开销比较

Tab. 2 Performance and overhead comparison of adaptive routing algorithms on Dragonfly

路由	解决问题①②③	空间复杂度	低逻辑开销	VC数量
UGAL[63]	× × ×	$O(1)$	√	4
PB[65]	√ × ×	$O(k)$	×	4
PAR[65]	√ × ×	$O(1)$	√	5
OFAR[67]	√√ ×	$O(1)$	√	2
PAR_{ph}[35]	√√ ×	$O(kT)$	×	5
ECtN[40]	√√ ×	$O(k^2)$	√	4
TPR[41]	√√ ×	$O(k^2T)$	×	4
Q-adaptive[38]	√√√	$O(k^4)$	×	5
DGB[66]	√√√	$O(k^4)$	×	4
UGAL-LE[44]	√ × ×	$O(k^3T)$	√	4

(×)。OFAR算法因使用哈密尔顿逃逸环和气泡流控的死锁避免机制故只需要2条VC,其余算法均通过逐跳VC升序的方式避免死锁。由于高的缓冲区开销、逻辑开销或需要额外的虚拟通道,除PAR算法的变体[35](PAR_{ph})外,其余自适应路由算法未被用于实际系统中。此外,一些Dragonfly网络的自适应路由算法还聚焦于缩短网络中非最小路径的长度[36, 45, 68-69],这与上述侧重于拥塞感知优化的路由算法是正交的。由于这类算法通常具有较高的逻辑开销且需要更多的缓冲区资源来存储非最小路径集,也未被用于实际系统。

UGAL在Dragonfly中的局限性也存在于其他如Flattened Butterfly、HyperX、Slim Fly等高阶网络中。在这类网络中,尤其是基于图论的或随机链路的不规则高阶网络,实现高效自适应路由还将具有更多挑战,严重限制了它们的实际部署。

5.2 各拓扑成本和性能的综合比较

表3按照拓扑的可拓展性比较了多种拓扑在构建中、大、超大系统下的性能和成本特征。基于低阶交换机、随机链路和数学图论等构造的非对称拓扑不包含在内。“二分带宽”指穿过最小割的总链路带宽与端点总注入带宽的一半之比。“每端点端口数”指每端点平均需要的路由器端口数量。端点与交换机之间的链路默认使用铜缆,但基于轨道优化的拓扑将使用光缆。Dragonfly、Dragonfly + 和Zettafly-3的组内链路使用铜缆,其余情况下交换机间链路默认使用光缆。HPN7.0组和Rail-Only网络的高带宽域的大小被设为8个端点。HPN7.0组在工程实现时设置了93.8%的二分带宽,但在表3的评估中统一设置为100%。成本和功耗评估使用了Zettafly[59]中的200 Gbit/s速率模型,对于400 Gbit/s、800 Gbit/s或更高速率的链路和交换机,其成本规律是相似的。其中200 Gbit/s铜缆和光缆的价格分别是246美元和1 350美元。交换机成本建模

表3 各拓扑成本与性能在构建不同规模系统下的比较

Tab. 3 Comparison of cost and performance across different topologies in building systems of varying scales

拓扑	规模	直径	网络可拓展性	二分带宽/%	无阻塞区域大小	每端点铜	每端点光缆	每端点端口数	每端点成本/美元	每端点功耗/W
两层胖树	中	2	$k^2/2$	100	$k^2/2$	1	1	3	3 087	49.3
两层BCube[21]		2	k^2	100	k^2	1	1	2	2 590	42.5
两层Rail-Only[52]		2	$4k^2$	100	$4k^2$	0	2	3	4 191	58.3
HPN7.0组[55]		2	k^2	100	k^2	0	3	4	4 343	61.3
两层ZCube[56]		1	k^2	100	k^2	1	2	4	2 655	45.5
三层胖树	大	4	$k^3/4$	100	$k^3/4$	1	2	5	5 431	71.8
3D HyperX[25]	超大	3	$k^4/256$	50	$k/4$	1.5	1	4	3 707	56.0
Dragonfly[5]		3	$k^4/64 + k^2/8$	50	$k^2/8$	2	0.5	4	3 155	51.5
Dragonfly +[31]		3	$k^4/16 + k^2/4$	50	$k^2/4$	2	0.5	4	3 155	51.5
Mesh-Tree[46]		6	$k^4/4$	50	$k^2/4$	1	2	5	5 431	71.8
Zettafly-3[59]		3	$k^4/64 + k^3/16$	50	$k^3/16$	2	1	5	4 327	62.8
Zettafly-4[59]		4	$k^4/16 + k^3/8$	50	$k^3/8$	1	2	5	5 431	71.8

为端口的线性函数,每端口 497 美元。此外,ZCube 和 HPN7.0 还将使用双端口网卡,根据 Colfax 网站数据[24],100 Gbit/s 的铜缆和光缆价格分别是 159 美元和 751 美元,每两个 100 Gbit/s 速率端口的成本被视为 1 个 200 Gbit/s 端口的成本。功耗模型由网卡、交换机和光模块组成。其中 200 Gbit/s单/双端口网卡功耗为 20 W,200 Gbit/s 路由器端口功耗为 6.75 W,200 Gbit/s 光模块功耗为 4.5 W,100 Gbit/s 光模块功耗为 3 W。

从表 3 的数据可以得出,每端点功耗和每端点成本之间大致呈线性正相关。当考虑构建中型系统时,基于双端口网卡的两层 ZCube 网络在保证高带宽和低直径的同时,提供了优异的成本效益。然而其可拓展性有限,基于 64 端口交换机网络仅支持 4 096 个端点。BCube 网络同样提供了良好的成本效益,然而 BCube 网络在路由实现上依赖端点转发流量,性能受限。当考虑构建大型系统时,三层胖树网络提供了良好的性能,然而网络成本效益和可拓展性表现一般。当考虑构建超大型系统时,Dragonfly 和 Dragonfly + 提供了优秀的可拓展性和成本效益,然而二者的无阻塞区域太小,组间带宽稀缺且网络性能依赖高效的自适应路由。Zettafly 同样具有较好的可拓展性,实现了介于胖树和 Dragonfly 的无阻塞区域大小和成本效益,同时支持有效的作业隔离和简单高效的自适应路由算法,因此,Zettafly 可能是未来超大规模高性能互连网络拓扑方案的首选。

表 3 仅给出了针对各类拓扑的静态理论分析结果,并未涉及其物理封装密度、网络容错能力,以及潜在的超额订阅策略等方面的探讨。拓扑结构的选型需要综合权衡多种因素,例如:路由器端口数量与机柜封装密度是否满足所设拓扑的实现要求;是否需要采用轨道优化设计,从而以增加光模块数量为代价提升负载均衡性能;各网络层级的规模应控制在何种范围;各层应配置何种比例的超额订阅。本文认为,最低成本或高性能高成本的拓扑方案未必是最佳选择。理想的拓扑结构应与运行的应用高度耦合,兼具成本效益与设计合理性,并在易于理解、封装与部署的同时,确保系统在运维层面具备较高的可管理性与稳定可靠性。

6　拓扑设计的挑战

6.1　成本挑战

成本是拓扑设计中最核心的考量因素,对设计方案的选择和优化具有显著的制约作用。网络的建设成本取决于路由器和链路尤其是光缆的数量。因此,如何在保证网络性能的同时减少光模块的数量是拓扑设计的关键目标。尽管基于图论的拓扑如 Slim Fly 等具有良好的摩尔界效率且需要更少的互连器件,但是这类拓扑难以被映射到合适的物理封装中。据了解,还未有基于图论的拓扑被大规模实际部署。相比之下,模块化与层次化的拓扑在设计阶段需充分考虑机柜的封装密度;更高的封装密度意味着更多链路可以在机柜内部完成封装,并可采用成本相对较低的电缆,从而实现性能与成本的更佳平衡。

6.2　可拓展性与低直径挑战

近期,博通推出了全球首款 102.4 Tbit/s 交换机芯片 Tomahawk 6[70],共有 1 024 个 100 Gbit/s Serdes 通道,可支持 256 个 400 Gbit/s 端口;相近地,英伟达也推出了支持 144 个 800 Gbit/s 端口的 Quantum-X800 IB[71] 交换机。使用这些先进的具有高端口数的路由器构建网络将显著简化拓扑设计。然而,大规模采购此类商用路由器的成本极为高昂。对于已具备路由器芯片自主设计能力的组织,例如国防科技大学“天河”超级计算机研制团队或美国 Cray 公司,通常更倾向于使用自研路由器而非最新商用产品。此外,由于知识产权授权、制程和先进封装工艺的限制,国产自研路由器芯片、整体带宽及端口数量通常低于国际顶尖水平。在此背景下,拓扑设计中面临着“大芯片小网络”与“小芯片大网络”权衡问题。如图 17 所示,受限的端口数量意味着需要构建更大规模的网络才能互连相同数量的端点。然而,网络规模的增加往往导致更大的直径和更高的端到端延迟。因此,在端口资源受限的条件下,如何构建既具高可扩展性又保持低直径的网络,仍然是当前拓扑设计中的重要挑战。

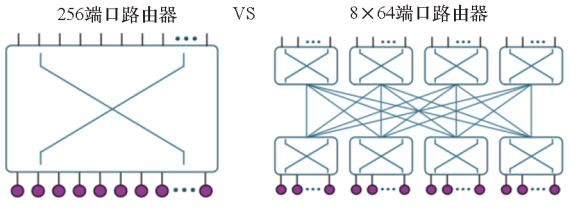


图 17　“大芯片小网络”和“小芯片大网络”的权衡

Fig. 17　Trade off between chip throughput and network size

6.3　路由挑战

拓扑与路由设计密切耦合、相辅相成,高效的路由算法能够充分发挥拓扑的潜在性能。然而,不同的拓扑结构对路由算法的复杂度要求并不相

同。例如,在胖树拓扑中实现流量的负载均衡相对容易,而在 Dragonfly 等最小路径相对稀缺的拓扑中则比较困难。后者由于存在间接路由等特性,需要依赖精确的自适应路由算法来确保性能。同时,路由算法的设计反过来也会对拓扑的选择产生约束。例如,当路由器芯片仅支持有限数量的虚拟通道,或无法在网络路由过程中动态切换数据包所使用的虚拟通道时,设计者往往更倾向于选择胖树等间接网络结构,而非 Torus 或 Dragonfly 等直接网络拓扑。这是因为直接网络通常需要更多虚拟通道,才能在保持路径多样性的同时避免死锁问题。此外,拓扑与路由设计还必须兼顾容错性。这要求路由算法在不额外增加虚拟通道的情况下,能够快速绕过失效的链路或交换机,从而保障网络的稳定性与可用性。

7 未来拓扑设计趋势

7.1 应用驱动型的拓扑设计

在传统的大型数据中心或超级计算机中,通常假设系统会运行多种混合的具有不同通信特征和带宽需求的任务,为了保证充足的网络性能,三层胖树通常是网络拓扑的首选。然而,HPC 和 AI 应用需求尤其是 LLM 训练需求的快速增长,推动着互连系统规模不断扩大,使得构建一个无收敛的胖树非常昂贵。如果选择一个低成本拓扑如 Dragonfly 可能无法满足 LLM 训练的高带宽需求。如果能够考虑系统运行的主要应用程序的通信特征,则可针对性地进行拓扑设计并将显著提升拓扑的性价比。例如 Zettafly 根据应用程序大小的分布特点,将网络划分为多个子网以提供更好的应用隔离。阿里巴巴为了避免 LLM 训练时广泛存在的"大象"流量在路由时潜在的哈希冲突问题,设计了双平面的 HPN 网络,并观察到超 96.3% 的 LLM 训练任务使用不到 1 000 块 GPU,则在组间连接中引入了 15∶1 的超额订阅,以优化成本。华为基于 LLM 训练的显著局部通信特征,提出了多维全连接的 UB-Mesh 网络。该设计主要依靠电缆连接,有效降低了部署成本,同时在高带宽和低延迟需求中保持了良好性能。

7.2 供电制约的拓扑设计

在后摩尔定律时代,数据中心的电力消耗正在朝着 GW 级迈进[1-2],电力供应将在部署超大规模数据中心时制约拓扑的设计。将系统的整个能耗(主要包括计算单元、网络以及散热设备的能耗)平摊到每块 GPU 上时,单 GPU 的能耗可达 1 200 ~ 1 500 W[19, 55]。这意味着构建一个 10 万卡 GPU 的集群可能需要 150 MW[19] 的能耗。该数值在许多情况下已超出单栋数据中心大楼的供电能力,具体情况还与当地的电力供应条件密切相关。未来大型数据中心集群的部署可能不再局限于单一建筑,而是部署在园区内的多栋大楼里,甚至是多个不同园区互连。然而,多模光模块支持的极限距离为 50 ~ 100 m[19, 55],大楼之间的连接将不得不使用极为昂贵的单模光模块,其价格大概是多模光模块的 3 倍。一种良好的设计是将拓扑设计与单栋建筑的供电能力进行耦合优化,例如阿里巴巴 HPN7.0 架构中,单个组可容纳约 15 000 块 GPU,与单栋建筑的供电上限相匹配;在多栋建筑之间,通过在组间部署具有高超额订阅比例的顶层交换机,并使用数量有限的单模光缆进行连接,从而在控制成本的同时实现大规模互连。

7.3 超节点内外拓扑的协同设计

在多种拓扑[21, 52, 54-55, 62]设计中均存在需要借助节点内网络进行路由转发的情况,而节点内和节点间网络涉及 PCIe、NVLink、IB/RoCE 等多种网络协议,导致流量转发时网络延迟更高。为缓解这一问题,领先厂商正着力推进统一的节点内网络协议,并构建包含更多 GPU 的超节点架构[72-73]。英伟达正与英特尔合作,致力于使用 NVLink 连接 x86 CPU 和 GPU[74],此举或将解决 CPU 和 GPU 间 PCIe 带宽不足的经典问题。同时,华为推出的 UB 总线从架构上实现了节点内与节点间网络协议的全面统一。统一的节点内网络协议与包含更多 GPU 的机架/机柜级超节点,有望成为未来数据中心发展的重要趋势。对于内部规模达到数百块 GPU 的超节点,其内部互连需要两级交换结构,从而引入更复杂的 Scale-Up 拓扑设计。这也意味着,未来的数据中心网络拓扑将不再仅仅围绕节点间的 Scale-Out 扩展,而是必须同时考虑节点内部与外部网络的协同优化设计。

8 结论

拓扑设计是高性能互连网络的核心,它决定了整体网络的性能与成本。近年来,HPC 超级计算机和 AI 数据中心的迅猛发展,推动着低延迟、高带宽、可扩展且具备成本效益的网络拓扑不断演进。总体而言,网络拓扑设计是一门平衡与折中的工程艺术,需要在多种相互制约的因素之间

寻找最佳方案,包括:建设成本、供电能力、路由器端口资源限制、虚拟通道数量限制、高效的自适应路由机制以及容错能力等。本文提出了拓扑设计的趋势,包括从运行的应用特点出发针对性地设计出具有高性价比的网络拓扑;从拓扑结构和大楼的供电能力协同设计出供电制约的网络拓扑;从超节点内趋向统一的网络协议设计出节点内外高效协同工作的网络拓扑。希望本文可为拓扑设计领域提供有意义的分析,并供未来相关系统设计者参考。

参考文献(References)

[1] ONTIVEROS J E, PATEL D, PENDEY A. AI training load fluctuations at gigawatt-scale-risk of power grid blackout[EB/OL]. (2025 - 06 - 26) [2025 - 11 - 15]. https://semianalysis.com/2025/06/25/ai-training-load-fluctuations-at-gigawatt-scale-risk-of-power-grid-blackout/.

[2] ONTIVEROS J E, PATEL D, ZHOU W, et al. xAI's colossus 2-first gigawatt datacenter in the world, unique RL methodology, capital raise[EB/OL]. (2025 - 09 - 17) [2025 - 11 - 15]. https://semianalysis.com/2025/09/16/xais-colossus-2-first-gigawatt-datacenter/.

[3] KIM J, DALLY W J, TOWLES B, et al. Microarchitecture of a high radix router [C]//Proceedings of the 32nd International Symposium on Computer Architecture (ISCA'05), 2005: 420 - 431.

[4] KIM J, BALFOUR J, DALLY W. Flattened butterfly topology for on-chip networks[C]//Proceedings of the 40th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO 2007), 2007: 172 - 182.

[5] KIM J, DALLY W J, SCOTT S, et al. Technology-driven, highly-scalable Dragonfly topology[C]//Proceedings of 2008 International Symposium on Computer Architecture, 2008: 77 - 88.

[6] LEI F, DONG D Z, LIAO X K, et al. Galaxyfly: a novel family of flexible-radix low-diameter topologies for large-scales interconnection networks [C]//Proceedings of the 2016 International Conference on Supercomputing, 2016: 24.

[7] LEI F, DONG D Z, LIAO X K, et al. Bundlefly: a low-diameter topology for multicore fiber[C]//Proceedings of the 34th ACM International Conference on Supercomputing, 2020: 20.

[8] LAKHOTIA K, BESTA M, MONROE L, et al. PolarFly: a cost-effective and flexible low-diameter topology [C]// Proceedings of SC22: International Conference for High Performance Computing, Networking, Storage and Analysis, 2022: 1 - 15.

[9] LAKHOTIA K, MONROE L, ISHAM K, et al. PolarStar: expanding the horizon of diameter-3 networks [C]// Proceedings of the 36th ACM Symposium on Parallelism in Algorithms and Architectures, 2024: 345 - 357.

[10] SINGLA A, HONG C Y, POPA L, et al. Jellyfish: networking data centers, randomly[C]//Proceedings of the 3rd USENIX Conference on Hot Topics in Cloud Computing, 2011: 12.

[11] FUJIWARA I, KOIBUCHI M, MATSUTANI H, et al. Skywalk: a topology for HPC networks with low-delay switches[C]//Proceedings of 2014 IEEE 28th International Parallel and Distributed Processing Symposium, 2014: 263 - 272.

[12] KOIBUCHI M, MATSUTANI H, AMANO H, et al. A case for random shortcut topologies for HPC interconnects[J]. ACM SIGARCH Computer Architecture News, 2012, 40(3): 177 - 188.

[13] BESTA M, HOEFLER T. Slim Fly: a cost effective low-diameter network topology [C]//Proceedings of SC14: International Conference for High Performance Computing, Networking, Storage and Analysis, 2014: 348 - 359.

[14] HAFNER P R. Geometric realisation of the graphs of McKay-Miller-Širáň[J]. Journal of Combinatorial Theory Series B, 2004, 90(2): 223 - 232.

[15] AL-FARES M, LOUKISSAS A, VAHDAT A. A scalable, commodity data center network architecture [J]. ACM SIGCOMM Computer Communication Review, 2008, 38(4): 63 - 74.

[16] WANG Y Y, DONG D Z, LEI F. MR-tree: a parametric family of multi-rail fat-tree[C]//Proceedings of 2021 IEEE International Performance, Computing, and Communications Conference (IPCCC), 2021: 1 - 9.

[17] NVIDIA. NVIDIA DGX SuperPOD: next generation scalable infrastructure for AI leadership[EB/OL]. [2025 - 11 - 15]. https://docs.nvidia.com/dgx-superpod/reference-architecture-scalable-infrastructure-h100/latest/_downloads/613ff7a85a665da4c3ff710b46eeec91/TEMD055-RA11333001-DSPH100-ReferenceArch.pdf.

[18] WANG Y Y, DONG D Z, LEI F. Understanding node connection modes in multi-rail fat-tree[J]. Journal of Parallel and Distributed Computing, 2022, 167: 199 - 210.

[19] PATEL D, NISHBALL D. 100,000 H100 clusters: power, network topology, Ethernet vs InfiniBand, reliability, failures, checkpointing[EB/OL]. (2024 - 06 - 18) [2025 - 11 - 15]. https://newsletter.semianalysis.com/p/100000-h100-clusters-power-network.

[20] GUO C X, WU H T, TAN K, et al. DCell: a scalable and fault-tolerant network structure for data centers [C]// Proceedings of the ACM SIGCOMM 2008 Conference on Data Communication, 2008: 75 - 86.

[21] GUO C X, LU G H, LI D, et al. BCube: a high performance, server-centric network architecture for modular data centers[C]//Proceedings of the ACM SIGCOMM 2009 Conference on Data Communication, 2009: 63 - 74.

[22] RIKEN Center for Computational Science. Fugaku[EB/OL]. [2025 - 11 - 15]. https://www.r-ccs.riken.jp/en/fugaku/.

[23] JOUPPI N, KURIAN G, LI S, et al. TPU v4: an optically reconfigurable supercomputer for machine learning with hardware support for embeddings[C]//Proceedings of the 50th Annual International Symposium on Computer Architecture, 2023: 82.

[24] Colfax International. COLFAX DIRECT[EB/OL]. [2025 - 11 - 15]. https://www.colfaxdirect.com/store/pc/home.asp.

[25] AHN J H, BINKERT N, DAVIS A, et al. HyperX: topology, routing, and packaging of efficient large-scale networks [C]//Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis, 2009: 1 - 11.

[26] Frontier. ORNL's exascale supercomputer is delivering world-leading performance in 2022 and beyond[EB/OL]. [2025 - 11 - 15]. https://www.olcf.ornl.gov/frontier/.

[27] Argonne National Laboratory. Aurora exascale supercomputer [EB/OL]. [2025 - 11 - 15]. https://www.anl.gov/aurora.

[28] Lawrence Livermore National Laboratory. EL Capitan: NNSA's first exascale machine [EB/OL]. [2025 - 11 - 15]. https://asc.llnl.gov/exascale/el-capitan.

[29] GIBSON D, HARIHARAN H, LANCE E, et al. Aquila: a unified, low-latency fabric for datacenter networks [C]// Proceedings of the 19th USENIX Symposium on Networked Systems Design and Implementation, 2022: 1249 - 1266.

[30] FENG G N, DONG D Z, ZHAO S Z, et al. GRAP: group-level resource allocation policy for reconfigurable Dragonfly network in HPC[C]//Proceedings of the 37th International Conference on Supercomputing, 2023: 437 - 449.

[31] SHPINER A, HARAMATY Z, ELIAD S, et al. Dragonfly +: low cost topology for scaling datacenters[C]//Proceedings of 2017 IEEE 3rd International Workshop on High-Performance Interconnection Networks in the Exascale and Big-Data Era (HiPINEB), 2017: 1 - 8.

[32] WEN K, SAMADI P, RUMLEY S, et al. Flexfly: enabling a reconfigurable Dragonfly through Silicon photonics [C]// Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, 2016: 166 - 177.

[33] Jülich Supercomputing Centre. JUPITER | The arrival of exascale in Europe [EB/OL]. [2025 - 11 - 15]. https://www.fz-juelich.de/en/jsc/jupiter.

[34] XIANG D, LI B, FU Y. Fault-tolerant adaptive routing in dragonfly networks[J]. IEEE Transactions on Dependable and Secure Computing, 2019, 16(2): 259 - 271.

[35] WON J, KIM G, KIM J, et al. Overcoming far-end congestion in large-scale networks[C]//Proceedings of 2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA), 2015: 415 - 427.

[36] RAHMAN M S, BHOWMIK S, RYASNIANSKIY Y, et al. Topology-custom UGAL routing on Dragonfly [C]// Proceedings of SC19: International Conference for High Performance Computing, Networking, Storage and Analysis, 2019: 1 - 15.

[37] MAGLIONE-MATHEY G, YEBENES P, ESCUDERO-SAHUQUILLO J, et al. Scalable deadlock-free deterministic minimal-path routing engine for InfiniBand-based Dragonfly networks[J]. IEEE Transactions on Parallel and Distributed Systems, 2018, 29(1): 183 - 197.

[38] KANG Y, WANG X, LAN Z L. Q-adaptive: a multi-agent reinforcement learning based routing on Dragonfly network[C]// Proceedings of the 30th International Symposium on High-Performance Parallel and Distributed Computing, 2021: 189 - 200.

[39] GARCÍA M, VALLEJO E, BEIVIDE R, et al. Efficient routing mechanisms for Dragonfly networks[C]//Proceedings of 2013 42nd International Conference on Parallel Processing, 2013: 582 - 592.

[40] FUENTES P, VALLEJO E, GARCÍA M, et al. Contention-based nonminimal adaptive routing in high-radix networks[C]// Proceedings of 2015 IEEE International Parallel and Distributed Processing Symposium, 2015: 103 - 112.

[41] FAIZIAN P, ALFARO J F, RAHMAN M S, et al. TPR: traffic pattern-based adaptive routing for Dragonfly networks[J]. IEEE Transactions on Multi-Scale Computing Systems, 2018, 4(4): 931 - 943.

[42] DE SENSI D, DI GIROLAMO S, HOEFLER T. Mitigating network noise on Dragonfly networks through application-aware routing[C]//Proceedings of SC19: International Conference for High Performance Computing, Networking, Storage and Analysis, 2019: 1 - 32.

[43] CHUNDURI S, HARMS K, GROVES T, et al. Performance evaluation of adaptive routing on dragonfly-based production systems[C]//Proceedings of 2021 IEEE International Parallel and Distributed Processing Symposium (IPDPS), 2021: 340 - 349.

[44] CHAULAGAIN R S, YUAN X. Enhanced UGAL routing schemes for Dragonfly networks[C]//Proceedings of the 38th ACM International Conference on Supercomputing, 2024: 449 - 459.

[45] BENITO M, FUENTES P, VALLEJO E, et al. ACOR: adaptive congestion-oblivious routing in Dragonfly networks[J]. Journal of Parallel and Distributed Computing, 2019, 131: 173 - 188.

[46] LI Y S, CHEN X H, LIU J, et al. OHTMA: an optimized heuristic topology-aware mapping algorithm on the Tianhe-3 exascale supercomputer prototype[J]. Frontiers of Information Technology & Electronic Engineering, 2020, 21(6): 939 - 949.

[47] GAN X B, TAN W. MT-lib: a topology-aware message transfer library for Graph500 on Supercomputers[EB/OL]. (2021 - 03 - 28) [2025 - 11 - 25]. https://arxiv.org/abs/2103.15024.

[48] YOUNG S, AKSOY S, FIROZ J, et al. SpectralFly: Ramanujan graphs as flexible and efficient interconnection networks [C]//Proceedings of 2022 IEEE International Parallel and Distributed Processing Symposium (IPDPS), 2022: 1040 - 1050.

[49] HOEFLER T, BONATO T, DE SENSI D, et al. HammingMesh: a network topology for large-scale deep learning[J]. Communications of the ACM, 2024, 67(12): 97 - 105.

[50] PATEL D, XIE M, NISHBALL D, et al. TPUv7: Google takes a swing at the King[EB/OL]. (2025 - 11 - 28) [2025 - 11 - 30]. https://newsletter.semianalysis.com/p/tpuv7-google-takes-a-swing-at-the.

[51] URATA R, LIU H, YASUMURA K, et al. Mission Apollo: landing optical circuit switching at datacenter scale[EB/OL]. (2022 - 08 - 22) [2025 - 11 - 30]. https://arxiv.org/abs/2208.10041.

[52] WANG W Y, GHOBADI M, SHAKERI K, et al. Rail-only: a low-cost high-performance network for training LLMs with trillion parameters [C]// Proceedings of 2024 IEEE Symposium on High-Performance Interconnects (HOTI), 2024: 1 - 10.

[53] AN W, BI X, CHEN G T, et al. Fire-Flyer AI-HPC: a cost-effective software-hardware co-design for deep learning[C]// Proceedings of the SC24: International Conference for High Performance Computing, Networking, Storage and Analysis, 2024: 1 - 23.

[54] ZHAO C G, DENG C Q, RUAN C, et al. Insights into

DeepSeek-V3: scaling challenges and reflections on hardware for AI architectures[C]//Proceedings of the 52nd Annual International Symposium on Computer Architecture, 2025: 1731 – 1745.

[55] QIAN K, XI Y Q, CAO J M, et al. Alibaba HPN: a data center network for large language model training[C]//Proceedings of the ACM SIGCOMM 2024 Conference, 2024: 691 – 706.

[56] YAN Z H, LI D, CHEN L, et al. From ATOP to ZCube: automated topology optimization pipeline and a highly cost-effective network topology for large model training[C]//Proceedings of the ACM SIGCOMM 2025 Conference, 2025: 861 – 881.

[57] Fujitsu Laboratories Ltd.. Fujitsu Laboratories develops technology to reduce network switches in cluster supercomputers by 40%[EB/OL]. (2014 – 07 – 15)[2025 – 11 – 30]. https://www.fujitsu.com/global/about/resources/news/press-releases/2014/0715 – 02.html.

[58] KATHAREIOS G, MINKENBERG C, PRISACARI B, et al. Cost-effective diameter-two topologies: analysis and evaluation[C]//Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, 2015: 1 – 11.

[59] DONG D Z, WANG Z Y, LEI F. Zettafly: a network topology with flexible non-blocking regions for large-scale AI and HPC systems[C]//Proceedings of the 52nd Annual International Symposium on Computer Architecture, 2025: 835 – 848.

[60] WANG R Q, DONG D Z, LEI F, et al. Roar: a router microarchitecture for in-network allreduce[C]//Proceedings of the 37th International Conference on Supercomputing, 2023: 423 – 436.

[61] WU K, DONG D Z, XU W X. COER: a network interface offloading architecture for RDMA and congestion control protocol codesign[J]. ACM Transactions on Architecture and Code Optimization, 2024, 21(3): 1 – 26.

[62] LIAO H, LIU B Y, CHEN X P, et al. UB-Mesh: a hierarchically localized nD-FullMesh data center network architecture[J]. IEEE Micro, 2025, 45(5): 20 – 29.

[63] SINGH A. Load-balanced routing in interconnection networks[D]. Palo Alto: Stanford University, 2005.

[64] FAANES G, BATAINEH A, ROWETH D, et al. Cray cascade: a scalable HPC system based on a Dragonfly network[C]//Proceedings of the SC '12: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis, 2012: 1 – 9.

[65] JIANG N, KIM J, DALLY W J. Indirect adaptive routing on large scale interconnection networks[C]//Proceedings of the 36th Annual International Symposium on Computer Architecture, 2009: 220 – 231.

[66] KASAN H, KIM G, YI Y, et al. Dynamic global adaptive routing in high-radix networks[C]//Proceedings of the 49th Annual International Symposium on Computer Architecture, 2022: 771 – 783.

[67] GARCÍA M, VALLEJO E, BEIVIDE R, et al. On-the-fly adaptive routing in high-radix hierarchical networks[C]//Proceedings of 2012 41st International Conference on Parallel Processing, 2012: 279 – 288.

[68] WANG Z H, WANG Q, LAI M C, et al. PIAR: path-improved adaptive routing for Dragonfly networks[C]//Proceedings of 2025 IEEE International Conference on Cluster Computing (CLUSTER), 2025: 1 – 11.

[69] BENITO M, VALLEJO E, BEIVIDE R. LIA: latency-improved adaptive routing for Dragonfly networks[J]. ACM Transactions on Architecture and Code Optimization, 2025, 22(1): 39.

[70] Broadcom Inc.. Broadcom ships Tomahawk 6: world's first 102.4 Tbps switch[EB/OL]. [2025 – 11 – 30]. https://www.broadcom.com/company/news/product-releases/63146.

[71] NVIDIA. NVIDIA Quantum-X800 InfiniBand switches[EB/OL]. [2025 – 11 – 30]. https://nvdam.widen.net/s/nfdzskhmnc/infiniband-datasheet-quantum-family-3231555.

[72] NVIDIA. NVIDIA GB200 NVL72: powering the new era of computing[EB/OL]. [2025 – 11 – 30]. https://www.nvidia.com/en-us/data-center/gb200-nvl72/.

[73] 华为云. 华为云发布 CloudMatrix 384 超节点 多项性能全面突破[EB/OL]. (2025 – 04 – 10)[2025 – 11 – 30]. https://www.huaweicloud.com/intl/zh-cn/news/20250424094932570.html.
Huawei Cloud. Huawei Cloud releases CloudMatrix 384 supernode, breaking through multiple performance aspects[EB/OL]. (2025 – 04 – 10)[2025 – 11 – 30]. https://www.huaweicloud.com/intl/zh-cn/news/20250424094932570.html.(in Chinese)

[74] NVIDIA. NVIDIA and Intel to develop AI infrastructure and personal computing products[EB/OL]. (2025 – 09 – 18)[2025 – 11 – 30]. https://nvidianews.nvidia.com/news/nvidia-and-intel-to-develop-ai-infrastructure-and-personal-computing-products.