综述

智算场景下集合通信库的挑战与发展趋势

高翔[1]，董斌[2]，肖晴[2]，简晟[1]，傅德基[1]
（1. 中国电信股份有限公司研究院，广东 广州 510630；
2. 中国电信股份有限公司上海分公司，上海 200120）

摘　要：随着人工智能（artificial intelligence，AI）大模型的发展，分布式训练对计算资源的需求显著增长，导致集群间通信的数据量大幅增加，给智算场景下的集合通信库带来了严峻的挑战。针对智算场景下计算任务的性能瓶颈以及业务需求，分析了当前集合通信库所面临的技术难题，同时也展望了集合通信库未来的发展趋势，比如开发更高效的通信算法、实现更加灵活的调度机制以及增强跨平台兼容性等，旨在为智能计算领域的研究与实际应用提供有价值的参考和支撑。

关键词：人工智能；大模型；智算场景；集合通信库
中图分类号：TP393
文献标志码：A
doi: 10.11959/j.issn.1000−0801.2025048

Challenges and development trends of collective communication libraries in intelligent computing scenarios

GAO Xiang[1], DONG Bin[2], XIAO Qing[2], JIAN Cheng[1], FU Deji[1]
1. Research Institute of China Telecom Co., Ltd., Guangzhou 510630, China
2. China Telecom Co., Ltd., Shanghai Branch, Shanghai 200120, China

Abstract: With the development of artificial intelligence (AI) large models progresses, the demand for computing resources in distributed training has increased significantly. This leads to a substantial increase in the amount of data communicated between clusters, which poses severe challenges to collective communication libraries in intelligent computing scenarios.. Focusing on the performance bottlenecks and business requirements of computing tasks in intelligent computing scenarios, the technical difficulties faced by current collective communication libraries were analyzed. At the same time, the future development trends of these libraries were also envisioned, such as developing more efficient communication algorithms, implementing more flexible scheduling mechanisms, and enhancing cross-platform compatibility, with the aim to provide valuable references and support for research and practical applications in the field of intelligent computing.

Key words: AI, large model, intelligent computing scenario, collective communication library

收稿日期：2024−12−01；修回日期：2025−01−21
通信作者：高翔，gaox15@chinatelecom.cn

0 引言

近年来，生成式大模型已成为一个极其热门的研究领域。从2022年出现的ChatGPT[1]到如今的OpenAI o1[2]模型，这类技术在计算机视觉、自然语言处理等多个科学领域取得了显著成就。随着数据量和模型规模的增长，模型训练与推理所需的计算资源也相应增加，这促进了大规模算力集群的发展与建设。同时，高效管理和利用集群中的计算节点成为影响大模型研究进展的关键因素，直接关系到模型迭代更新的速度及其实际部署的效果。

集合通信库是当前分布式大模型和大规模并行计算中广泛采用的通信技术，主要负责计算节点内部及节点间的数据同步。集合通信库提供了多种通信原语，包括广播、聚合和散播等，并支持多种集群网络拓扑结构。通过优化通信算法和选择合适的网络拓扑，集合通信库能够显著地提高多个计算节点的通信效率，减少时间开销，从而满足智算场景对高吞吐量、高容错性和低时延的要求。这不仅提高了芯片计算资源的利用率，还缩减了大模型的训推时间，增强了实际应用中的系统稳定性。

目前智算集群中心正朝着算力规模扩张、通信网络优化和异构硬件兼容等多个方向发展[3]。为适应智算环境的变化，集合通信库将面临多种机遇和挑战，因此，研究智算场景下集合通信库的挑战与发展趋势具有重要意义。本文系统回顾了集合通信的相关知识以及主要的集合通信库，从多个维度分析现有集合通信库存在的问题，并着重探讨未来的发展方向，为相关领域的研究人员提供高层视野，助力人工智能技术和应用的持续进步。

1 集合通信库技术原理与现状

1.1 集合通信基本概念

集合通信（collective communication）是并行计算和分布式系统中一种重要的通信模式[4]，用于多个进程或节点之间的数据交换和同步。在高性能计算（high performance computing，HPC）和人工智能（artificial intelligence，AI）智算领域，集合通信是不可或缺的技术基础，尤其体现在处理海量数据流以及训练生成式大模型的过程中。集合通信的主要目的是通过拓扑通信算法和数据同步机制，确保参与同步的各个节点并行一致地完成计算任务，并且最大化整体计算效率，减少通信开销。

集合通信在智算场景中的作用如下。

（1）提供各种通信原语以实现多节点数据的聚合及传播，支撑大模型分布式训练中的梯度同步、参数更新等关键步骤。

（2）通过合理的拓扑结构和通信算法设计，减少集群通信开销，提高资源的利用率和模型训练速度。

（3）集成了调度优化策略及拓扑感知算法，增强智算系统的可扩展性和容错性，保持大模型计算的稳定性。

（4）简化通信管理操作的复杂性，提供简洁抽象的编程接口，为大模型算法开发屏蔽底层通信细节。

1.2 集合通信原语

集合通信库通常遵循消息传递接口（message passing interface，MPI）通信标准，提供了一组标准化的通信原语，用于实现跨进程的数据传输和同步操作。这些原语包括Broadcast、Gather、Scatter、Reduce、All-Reduce、All-Gather、Reduce-Scatter、All-to-All等，通信原语操作示例如图1所示。

集合通信率常见通信原语对比见表1，其中，Broadcast将特定进程或节点的数据传播至系统中的其他节点，确保所有节点最终拥有相同的数据副本。在智算场景中，Broadcast常用于大模型权重参数的初始化以及模型更新后的

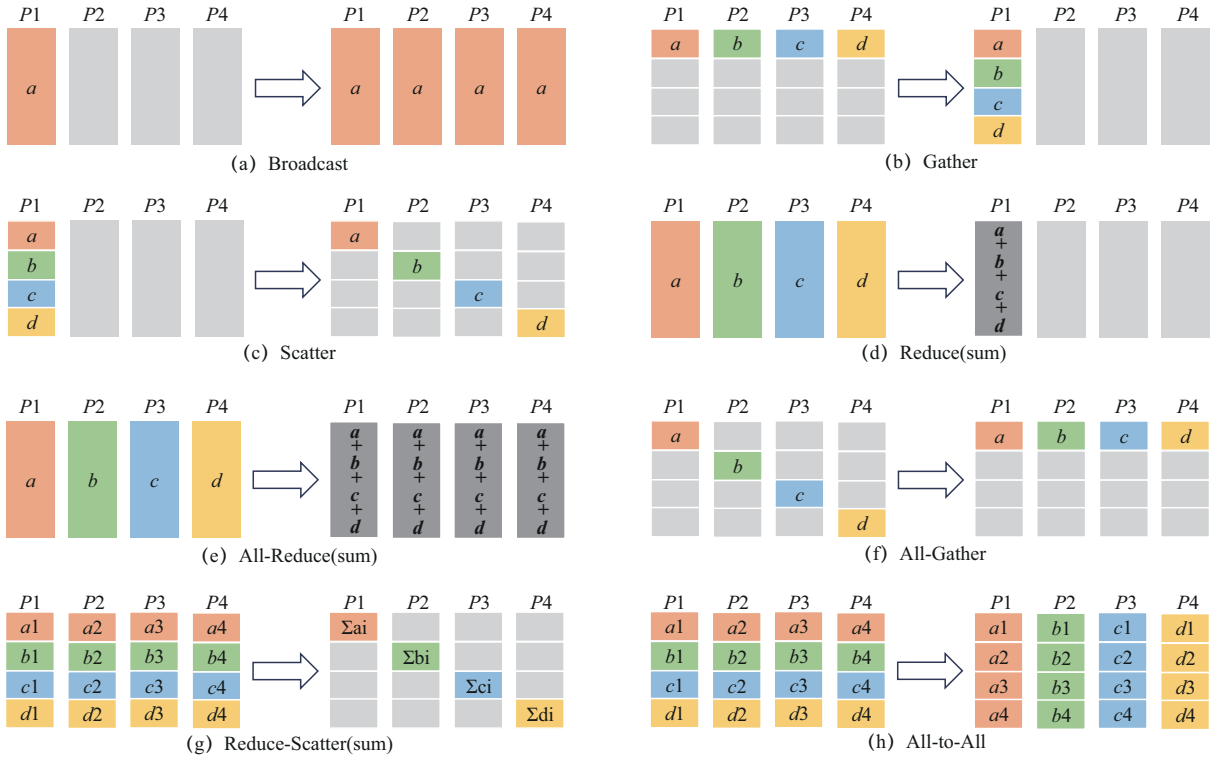

图 1　通信原语操作示例

表 1　集合通信库常见通信原语对比

通信原语	功能描述	特点	适用场景
Broadcast	将一个节点的数据发送到所有其他节点	数据一致性、简单高效	参数初始化、权重同步
Gather	将所有节点的数据收集到一个主节点	数据集中处理、维度拼接	前向计算结果汇总
Scatter	将主节点的数据分割后分发到其他节点	数据分配、反向操作于 Gather	模型权重分配
Reduce	对所有节点的数据执行规约操作（如求和、求最大值等）并将结果发送到主节点	数据聚合、支持多种运算	梯度聚合、参数更新
All-Reduce	对所有节点的数据执行规约操作并将结果广播到所有节点	数据全局同步、高效并行处理	反向传播梯度同步
All-Gather	每个节点收集所有其他节点的数据	数据全局共享、类似 Gather+Broadcast	张量并行处理
Reduce-Scatter	对所有节点的数据执行规约操作后分割并分发到各节点	优化通信效率、减少数据传输量	优化 All-Reduce 操作
All-to-All	每个节点与其他所有节点交换数据	数据完全交换、矩阵转置	并行计算中的矩阵操作

权重同步。如图 1（a）所示，主进程 P_1 进行 Broadcast 广播后，P_2、P_3、P_4 均保存了 P_1 发送的数据。

Gather 操作则是将所有节点的数据收集至一个主节点，并将各节点数据按照节点总数进行维度拼接，存储于主节点。在大模型的张量并行处理中，Gather 可用于汇总所有节点的前向计算结果。如图 1（b）所示，每个进程 P_i 都存储着不同的数据，通过 Gather 操作后，不同的数据按维度拼接后存储于主进程 P_1。

Scatter 与 Broadcast 操作截然相反，首先将主节点的数据按节点总数分割，然后将每个分割的

数据块发送至对应的节点。在大模型的流水线并行处理中，Scatter可用于将初始化的模型权重分配至各个节点。如图1（c）所示，主进程P_1按照总进程数将数据分为a、b、c、d这4块，并根据进程序号发送到对应进程P_i上。

Reduce操作也称为规约运算，涉及对所有节点的数据执行简单运算，包含Sum、Product、Max、Min、Mean等，并将结果聚合至主节点。例如，在分布式训练中，Reduce用于聚合各个工作节点数据，如图1（d）所示，4个进程分别持有不同的数据，经过Reduce操作后，仅有主进程P_1存储了各进程数据的求和结果。

All-Reduce操作则是对所有节点的数据执行规约运算，并将结果广播至所有节点，类似于先执行Reduce再执行Broadcast，以确保数据在所有节点间的同步，在大模型训练的数据并行处理中主要用于同步各个显卡上的反向传播梯度。如图1（e）所示，所有进程的数据求和得到$a+b+c+d$，并将结果发送给了每个进程P_i。

All-Gather操作允许多个节点将各自的数据分发至其他节点，类似于在Gather操作后将主节点的数据广播至所有节点，使得每个节点都能获得其他节点的数据。在张量并行处理中，All-Gather用于收集不同显卡上的模型参数部分，并同步至所有设备，以便统一输出前向计算结果。如图1（f）所示，各进程通过All-Gather操作保存了全部数据，即a、b、c、d按维度拼接的结果。

Reduce-Scatter操作是将所有节点的数据按维度进行规约运算，然后将结果根据节点数量分割并发送至对应的节点，可以视为Reduce和Scatter的组合。这一操作可用于优化All-Reduce操作的通信效率。如图1（g）所示，各进程经过Reduce操作获得中间结果$(\sum a_i, \sum b_i, \sum c_i, \sum d_i)$，在通过Scatter操作按顺序将中间结果分发给进程P_i。

All-to-All操作允许每个进程与其他所有进程交换数据。具体来说，每个节点的数据首先按节点总数分割，然后将分割的数据块发送至对应序号的节点。根据每个节点发送和接收的数据量是否相等，All-to-All操作可分为均匀和非均匀2类，常用于并行计算中涉及矩阵转置的场景。如图1（h）所示，进程P_i的数据划分为a_i、b_i、c_i、d_i这4个子块，通过All-to-All操作后，进程P_1、P_2、P_3、P_4分别存储a_i、b_i、c_i、d_i的数据，行为上等效于线性代数中的矩阵转置。

1.3 集合通信常用网络拓扑

集合通信的网络拓扑是分布式系统中节点间数据传输的基础架构，确立了节点的连接规则，对通信效率和系统可扩展性具有决定性影响。选择合适的拓扑结构对于提升带宽利用率、降低时延以及优化智能计算任务至关重要。一旦拓扑结构确定，便可运用特定算法，充分发挥网络结构的优势，从而增强通信性能。常见的拓扑结构包括Ring、Torus、Fat-Tree和Mesh等，通信网络拓扑如图2所示。

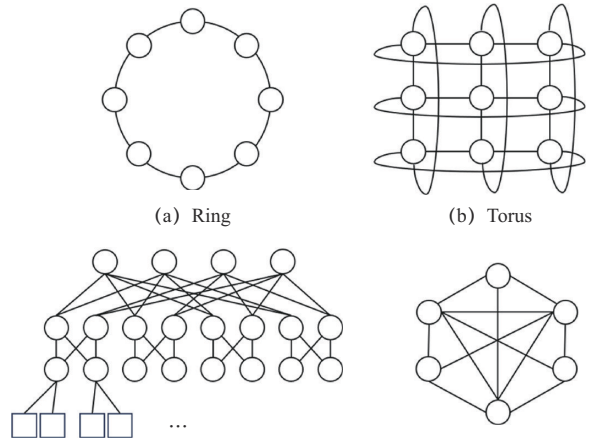

图2 通信网络拓扑

Ring拓扑[5]，作为网络通信领域的一种经典模式，其核心在于构建一个闭环的节点网络，每个节点均与左右相邻节点相连，形成数据的发送与接收链路，数据在环形路径上单向流动，经过一系列中间节点后抵达目标。其优势在于结构简洁、数据流向单一、有效规避通信冲突、传输稳

定高效；节点依序转发数据，不需要复杂路由决策，简化通信流程；网络扩展便捷，新增节点成本低且无须中心协调，布线简易。但存在单点故障风险，一旦节点失效会致网络瘫痪，特定情形下数据遍历环形路径会引发时延与拥塞，降低传输灵活性。

相比之下，Torus 拓扑[6]则是一种更为复杂且多维的网络结构。其由多个节点环面组成，每个节点在多维空间中与 2 个邻近节点相连，形成丰富的连接网络。当维度降至一维时，Torus 拓扑便退化为 Ring 拓扑。在设备量较大时优势显著，可缩短环面通信路径、提升网络容量、降低设备接入成本；提供多元通信路由，保障高带宽、高容错与低时延，有效均衡负载。然而，维度增加会使网络规模与布线复杂度剧增，系统扩容也会导致通信跳数上升引发时延问题，在设计部署时需要谨慎权衡利弊。

Fat-Tree 拓扑[7]则是一种以交换机为中心的树状网络结构，旨在解决传统树状拓扑的通信瓶颈问题。其分为核心层、汇聚层和接入层 3 层，其中，核心层和汇聚层由高性能交换机构成，负责数据的高速转发；接入层则连接集群服务器，为终端用户提供服务。Fat-Tree 拓扑的优势在于其高效的通信性能和良好的容错性，此外，Fat-Tree 拓扑的递归可扩展性和可划分性使得其能够设计多种高效通信算法，满足不同应用场景的需求。然而，核心与汇聚层对带宽要求苛刻，需要高性能交换机支撑，成本高昂；叶子节点间通信路径多，增加了调度与负载均衡难度，设计部署时需要综合考量多方面因素。

Mesh 拓扑[8]是一种高度互联、去中心化的网络结构，具体可划分为两大类别：一类是全互联 Mesh 拓扑，在此架构下，任意 2 点间均可实现直接连通；另一类则是部分互联 Mesh 拓扑，通过若干中继节点，将较远的节点连接起来。Mesh 拓扑的显著优势首先体现在数据传输的速率与可靠性上，全互联 Mesh 下任意节点间通信仅需一跳，数据传输快速可靠；去中心化设计使网络抗毁性强，节点故障不影响整体运行；扩展性良好，增删节点简便。但集群部署需要大量硬件投入，运营维护复杂烦琐，对资源与人力要求较高。通信库常见网络拓扑对比见表 2。

在分布式推理及训练系统中，网络拓扑的选择对效率和性能至关重要，如表 2 所示，每种结构都具备独特的特性和适用环境，能够适应不同的硬件条件和通信需求：Ring 拓扑简单便宜，适合小型应用，但扩展性和时延差；Torus 拓扑通过多维互联提升性能，适合中大型集群和高性能计算；Fat-Tree 拓扑通信高效、容错强，适合数据中心大规模训练，但成本高；Mesh 拓扑灵活冗余，可靠性极高，但部署和维护要求高。选择合适的拓扑结构应综合考虑具体需求、预算及性能要求。

1.4　集合通信常用算法

集合通信算法是对通信原语的具体实践形式，详细阐明了多个进程或节点间如何同步数据、协调操作的系列准则。这些算法能够促使一组进程协同参与特定的通信模式，诸如数据的发

表 2　通信库常见网络拓扑对比

拓扑结构	特性	优点	缺点	适用场景
Ring	单向闭环，节点依次连接	简单易实现，成本低	单点故障风险高，时延随节点增加而提高	小型或成本敏感场景
Torus	多维环形，节点多方向连接	通信路径短，容错性好	布线复杂，调度复杂	中大型高性能计算集群
Fat-Tree	树状结构，分层设计	低时延，高容错性	高带宽需求，部署成本高	数据中心大规模分布式任务
Mesh	网状结构，节点高度互联	高可靠性，低时延	硬件成本高，维护复杂	高可靠性需求场景

送、接收及聚合等，并且能与相应的抽象网络拓扑相结合，从而圆满达成既定任务。在智能计算场景中，诸如Ring、Halving-Doubling、Binomial Tree、Bruck等通信算法均得到了广泛运用。

Ring算法[9]的核心构思在于，将集群内的所有节点构建成一个逻辑上的环状结构，数据在这个环状结构中以单向的方式在各节点间流动，节点接收处理后传递给下一节点，直至所有节点都完成了数据的传输与处理，并且各自收到了所需的数据全集。Ring算法在All-Gather和All-Reduce操作中尤为常见，假设总节点数为p，则All-Gather操作需要$p-1$步数据传输，而All-Reduce操作需要$2(p-1)$步。Ring算法的优势在于其通信复杂度相对较低，且易于实现负载均衡，但处理小数据量通信的效率略显不足，并且随着节点数量的增多，时延也会呈现出线性增长的趋势。

Halving-Doubling算法[10]是一种广泛应用于超大规模集群训练中的集合通信算法，适合于实现All-Reduce操作。该算法通过多次将距离相隔2的k次方（k为非负整数）的节点进行两两配对分组，使得每组内的节点能够相互传输数据。随着通信的进行，每次传输的数据量逐渐减少，最终实现了所有节点数据的规约。在总节点数为p的假设下，Halving-Doubling算法所需的数据传输步数仅为$\mathrm{lb}\,p$。相较Ring算法，Halving-Doubling算法通信步数更少，有助于显著降低通信时延，更加适用于大规模集群场景。但算法后期可能由传输数据量大引发网络拥塞，且节点数非2的幂次时需要额外处理，增加了算法复杂性。

Binomial Tree算法[11]将节点组织为二叉树结构实现数据流通，数据分发与汇聚递归进行。Binomial Tree算法在Broadcast和Reduce操作中表现出色。假设集群的总节点数为p，在执行Broadcast操作时，数据的传输步数为$\mathrm{lb}\,p$；在执行Reduce操作时，数据的传输步数则是$2\mathrm{lb}\,p$。该算法能充分利用带宽，擅长短消息与小规模集群通信，但节点度受限，高节点度场景适用性不佳。

Bruck算法[12]是一种基于循环位移与按位交换操作的通信模式，数据在节点队列旋转传输，通信发生在多组间隔节点且间隔以2的幂次递增。Bruck算法在All-Gather和All-to-All操作中有着广泛的应用。假设集群的总节点数为p，Bruck算法进行通信所需的次数为$\mathrm{lb}\,p$。其优势是对节点数无特殊要求，非标准节点数与短消息场景下通信开销低，但内存移动开销大、算法实现复杂。

通信库常见通信算法对比见表3。在实际应用中，需要综合考量集群规模、任务特性及网络环境等因素选择合适算法。小型快速响应集群可优先考虑Binomial Tree算法；大规模集群下Halving-Doubling算法因其高效通信步数更具优势；Bruck算法适用于非标准节点数场景；Ring算法则以实现简便见长。

1.5 主流集合通信库分析

集合通信库是高性能多GPU/NPU计算架构

表3 通信库常见通信算法对比

算法名称	原理简述	优势	劣势	适用场景
Ring	节点成环，数据单向流动处理	复杂度低、易均衡负载	小数据慢、节点多、时延高	中小规模数据并行，节点数适中
Halving - Doubling	按2的幂次距离配对节点规约数据	通信步少、适合大集群	后期拥塞、非2幂次复杂	大规模模型训练数据并行
Binomial Tree	节点组成二叉树传输数据	利用带宽、小集群短消息优	节点度受限	小规模集群短消息处理
Bruck	循环位移与按位交换，间隔2的幂次通信	不限节点数、短消息开销低	内存开销大、实现复杂	节点数不规则且短消息场景

中的核心要素，其构建了一套标准化的信息交换接口，旨在解决并行计算环境下不同进程间的通信挑战。在AI、HPC等大型计算通信场景中，主流的集合通信库凭借其优化的多节点数据传输能力，显著提升了数据同步效率，缩短了计算周期，进而增强了系统整体性能。本节将概述多种现有的集合通信库，详细剖析其特性，主流集合通信库特性对比见表4。

NVIDIA集合通信库（NVIDIA collective communications library，NCCL）是业内非常重要且被广泛使用的集合通信库，专为NVIDIA GPU架构及网络环境打造，旨在多GPU及多节点系统中实现高效数据传输与协同处理。针对节点内的PCIe和NVLink链路，以及跨节点的NVIDIA Mellanox网络，NCCL提供了一系列优化的集合通信操作，确保了数据传输的高带宽和低时延。其核心优势在于自动拓扑识别能力，能智能地在与NVIDIA相关的硬件环境，如PCI Gen4及IB HDR等上寻找最优通信路径。NCCL可与NVIDIA SHARP技术集成，SHARPV2能显著提升数据传输性能，助力All-Reduce等操作。在节点间通信方面，NCCL兼容InfiniBand verbs、libfabric、RoCE及IP Socket等多种技术，以适应多变的网络环境。同时，其动态路由功能利用InfiniBand动态路由机制重新分配流量，有效地解决了端口拥堵问题。NCCL已被多个主流深度学习框架集成，包括PyTorch、MXNet、Caffe2、Chainer及TensorFlow，为加速深度学习模型的训练过程提供了有力支持。

华为集合通信库（Huawei collective communication library，HCCL）是专为华为昇腾AI处理器设计的高性能通信解决方案，旨在满足单机多卡及多机多卡环境下的数据并行与模型并行需求。该库集成了All-Reduce、Broadcast等多种通信原语，并可在HCCS、RoCE、PCIe等多种通信链路上运行，同时兼容Ring、Mesh、HD、NHR等多种通信算法。在硬件拓扑方面，昇腾硬件服务器采用Full Mesh内部连接，而集群多节点则常用Fat-Tree拓扑结构。HCCL能够智能识别当前拓扑，并自动选择最优的通信算法，同时引入计算通信统一调度机制，以降低调度成本，提高硬件资源的使用效率。此外，HCCL还提供了一套高效的计算通信并行处理方案，使计算任务和通信任务能够同步执行，从而显著缩短整体任务时

表4　主流集合通信库特性对比

集合通信库	NCCL	HCCL	RCCL	Gloo	oneCCL	MSCCL
厂商	英伟达	华为	AMD	Meta	英特尔	微软
目标硬件	NVIDIA GPU	昇腾NPU	NVIDIA GPU AMD GPU	通用GPU	Intel芯片	通用GPU
差异	针对NVIDIA GPU调优，集合通信标杆	基于昇腾NPU的通信加速	支持AMD GPU，易从英伟达生态迁移	为CPU/GPU之间通信加速	为英特尔异构硬件加速	提供自定义通信算法能力
点对点通信	是	是	是	是	是	是
异构通信	否	否	否	是	是	否
拓扑感知	是	是	是	是	是	是
故障检测	是	是	是	是	是	是
通信算法	（1）Double-Binary-Tree （2）Ring	（1）Ring （2）Mesh （3）Havling-Doubling （4）Pair （5）Star	（1）Ring （2）Tree	（1）Ring （2）Ring-Chunked （3）Halving-Doubling （4）BCube （5）Pairwise-Exchange	（1）Ring （2）Tree	（1）Ring （2）All-Pairs （3）Hierarchical （4）Two-Step

间。作为CANN框架的重要组成部分，HCCL不仅全面支持CANN生态下的深度学习框架MindSpore，还与常用的PyTorch和TensorFlow框架兼容，为国产AI领域的发展提供了全面的生产支持。

ROCm集合通信库（ROCm communication collectives library，RCCL）是一个专为AMD GPU设计的高性能集合通信库，旨在提升大规模并行计算中的通信效率，并且能支持迁移至NVIDIA GPU环境。其支持All-Reduce、Broadcast等集体通信操作，以及GPU间的点对点通信，可应用于XGMI、PCIe等通信链路，同时兼容InfiniBand verbs、RoCE、IP Socket等多种技术。RCCL利用Ring和Tree算法，结合拓扑感知、高速互连和RDMA技术，实现高带宽、低时延的通信。ROCm软件库包含RCCL通信组件，为深度学习开发提供了一站式的生态系统，支持PyTorch、TensorFlow和JAX等框架。

Gloo是Meta公司专为机器学习打造的集体通信库，提供Barrier、Broadcast、All-Reduce等多种算法，广泛应用于分布式深度学习训练中。Gloo具备数据传输的抽象化能力，能在IP Socket、InfiniBand、RoCE等不同网络环境间灵活切换。特别是当使用InfiniBand时，借助NVIDIA的GPU Direct技术，Gloo能实现GPU内存的直接传输，减少数据复制，提升通信效率。在通信算法方面，Gloo集成了Ring、Ring-chunked、Halving-Doubling和BCube等多种实现，并兼容CUDA和NCCL，增强了算法实现和拓扑感知能力。在应用层面，Gloo支持TensorFlow、PyTorch、MXNet和Keras等多种深度学习框架。

oneAPI集合通信库（oneAPI collective communications library，oneCCL）是英特尔专为深度学习模型训练设计的高效通信库，其基于Intel MPI Library和libfabrics构建，支持InfiniBand、Cornelis Networks、Ethernet等多种网络互连标准。oneCCL采用先进的通信算法，确保在各种硬件平台上都能高效地运行。它支持SYCL，并能与Level Zero驱动层无缝对接，从而充分利用Intel CPU和GPU等现代硬件的性能潜力，支持操作的异步和乱序执行，并能根据需求灵活使用一个或多个核心来优化网络使用。目前，oneCCL已集成到Horovod和PyTorch等流行的分布式训练及机器学习框架中，助力其进行高效的并行计算。

微软集合通信库（Microsoft collective communication library，MSCCL）是一个专为Microsoft Azure上的异构加速器设计的平台，用于执行自定义集合通信算法。MSCCL建立在NCCL之上，并利用其构建块来执行用户自定义的集合通信算法。MSCCL致力于提供一个统一、高效且可扩展的框架，以跨多个加速器执行集合通信算法，其关键特性之一是可编程性。由于加速器之间的互连具有不同的时延和带宽，通用的集合通信算法可能并不适用于所有拓扑和缓冲区大小。因此，MSCCL允许用户针对特定的拓扑结构和缓冲区大小编写高度优化的集合通信算法。其包含一个领域专用语言和一个编译器，用于生成MSCCL运行时可以执行的中间表示。目前，MSCCL支持NVIDIA和AMD GPU，由于与NCCL类似，因此可以替换PyTorch框架中的默认集合通信方式。

总体而言，各集合通信库特性各异，主要区别在于硬件平台支持、通信操作优化及API/工具集。NCCL专为NVIDIA GPU优化，兼容多网络环境，集成于主流深度学习框架；HCCL针对昇腾AI处理器，降低调度成本，适用国产昇腾AI生态；RCCL服务于AMD GPU，兼顾跨平台兼容性和通信效率，支持细粒度事件追踪；Gloo由Meta开发，提供灵活的数据传输抽象层，适合不同网络环境，尤其在InfiniBand下表现优异；

oneCCL 基于 Intel 硬件，支持异步执行和低精度操作，优化内存使用与计算性能；MSCCL 则以高度可编程性著称，允许用户编写定制化通信算法，适应特定拓扑结构。每种库各有侧重，适用于不同的硬件平台和应用场景，共同推动了分布式计算的发展。

2　智算场景下集合通信库的技术挑战

随着 AI 技术的持续革新，深度学习模型尤其是大规模 AI 模型的训练与推理，对计算资源的渴求呈爆发式增长。智算中心作为关键支撑设施，集成高性能计算节点与前沿通信技术，然而集合通信库在此场景下却面临诸多棘手难题。

2.1　算力规模与通信效率的双重挑战

深度学习模型规模的不断扩张，算力需求呈现出前所未有的指数级增长态势，导致计算节点数量激增。据估算，GPT-3 的训练就动用了数千块高端 GPU[13]。在此背景下，节点间的通信开销在系统总计算时间中的占比显著上升，成为制约系统整体性能提升的关键因素。

为应对大模型训练中的硬件限制，研究者们提出了多种分布式策略，包括数据并行、张量并行和流水线并行等。典型 AI 训练并行计算模式如图 3 所示，数据并行通过将数据集分割成多个子集，并在不同节点上并行处理，有效地提高了训练速度，但伴随而来的大量参数同步操作（如 All Reduce）对通信带宽和时延提出了更高要求；张量并行则将模型参数分割到不同节点上，减少了单个节点的内存负担，但引入了复杂的 All Gather 通信模式；流水线并行通过将模型切分为多个阶段，在不同节点上串行执行，虽然降低了同步需求，但阶段间的数据传输仍是一大挑战[14]。

随着大模型向多模态、长序列、混合专家架构的演进，分布式策略变得更加复杂。例如，序列并行通过分割输入序列，使不同节点处理序列的不同部分，而专家并行则在模型中引入多个专家网络，根据输入动态选择执行，这 2 种策略均显著增加了 All Gather 和 All-to-All 等集合通信操作的需求。这些操作要求计算设备间具备超低时延、超高带宽的通信能力，对单个计算节点及整个网络的性能提出了更高的要求。不同并行模式下的通信需求见表 5。

总体而言，智算场景下，集合通信库面临着通信算力规模与通信效率的双重挑战。

（1）通信时延问题：复杂的网络拓扑结构和庞大的通信数据量使得节点间的通信时延成为一大难题。在分布式深度学习中，梯度同步的时延会直接影响模型训练速度，甚至对模型的收敛效果造成负面影响。例如，ZeRO++ [15]是结合量化

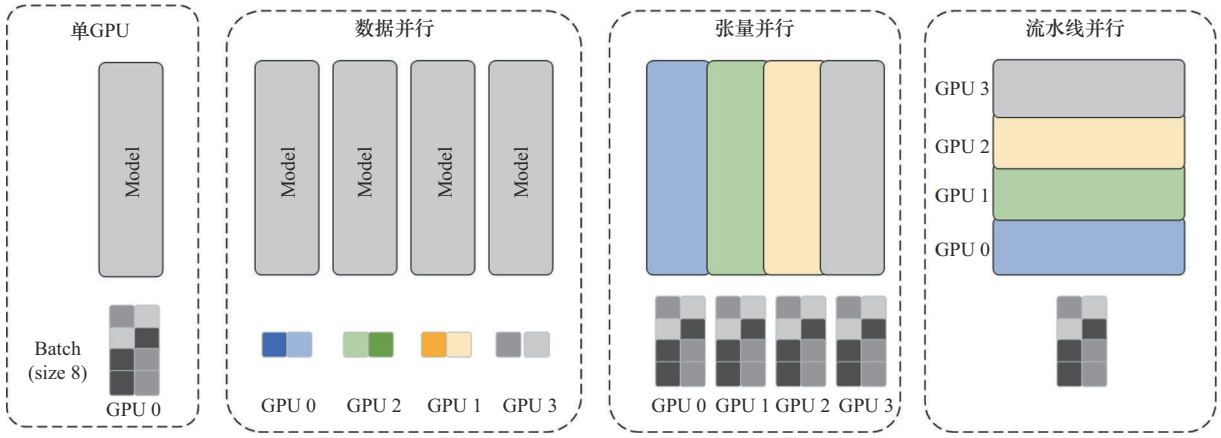

图 3　典型 AI 训练并行计算模式

表5 不同并行模式下的通信需求

类型	操作	节点规模	数据量	备注
张量并行TP	All Reduce、All Gather、Reduce Scatter	2/4/8 Rank	MB-GB	计算通信可以隐藏，主要是节点内并行，不适合跨节点
流水并行PP	Send/Recv	2 Rank	KB-MB	通过Micro Batch实现计算通信重叠
数据并行DP	All Reduce、Broadcast	理论无限	GB	计算通信重叠，可隐藏流水，节点规模受线性度影响
专家并行EP	All-to-All	理论无限	KB-MB	计算通信串行，不可隐藏流水

和节点内带宽的通信优化方案，能够使用低精度量化技术显著减少通信量，可以将强化学习中的人类反馈训练速度提高3.3倍[16]。而在大规模分布式系统中，复杂的网络拓扑结构（如树状、环状、网状等）和庞大的通信数据量共同导致了节点间通信时延的增加。梯度同步的时延直接影响模型训练速度，严重时可能导致模型收敛速度减慢甚至训练失败，例如，在同步SGD（stochastic gradient descent）算法[17]中，通信时延成为制约训练效率的关键因素。

（2）通信带宽瓶颈：尽管高速网络接口技术（如InfiniBand EDR/HDR、RDMA等）的普及提升了网络带宽，但在大规模节点集群中，带宽资源的有效分配和调度仍面临挑战。带宽竞争、网络拥塞以及不公平的带宽分配策略都可能降低通信性能，进而影响整体计算效率，特别是在采用密集通信操作的分布式策略中，带宽瓶颈尤为突出。

2.2 异构硬件兼容性的挑战

在智算场景下，计算节点的硬件配置呈现出前所未有的多样性，涵盖了从高性能CPU到各类加速卡（如GPU、NPU、FPGA）的广泛范围。这种硬件异构性不仅为高性能计算提供了强大的算力支持，同时也对集合通信库的兼容性提出了更为严峻的挑战。

（1）通信协议差异与兼容性挑战：在异构计算环境中，不同硬件平台支持的通信协议和接口标准存在显著差异。例如，NVIDIA GPU通常依赖于CUDA通信库（如NCCL）来实现高效的数据交换，而FPGA则可能依赖于特定的硬件描述语言（HDL）或厂商专有的接口进行通信。这种协议多样性导致集合通信库在实现过程中必须处理复杂的兼容性问题，以确保在不同硬件平台上都能实现高效、稳定的数据传输。

为应对这一挑战，学术界和工业界提出了多种解决方案。一种有效的方法是构建通信协议的抽象层，该层能够屏蔽底层硬件的具体协议差异，为上层应用提供统一的通信接口。例如，Open消息传递接口（message passing interface，MPI）[18]作为广泛使用的并行编程模型，通过其底层的通信协议抽象层，支持了多种硬件平台上的高效通信。此外，插件化架构也是解决协议兼容性问题的一种有效手段，允许集合通信库根据实际需要动态加载和配置相应的通信协议模块，例如，无问芯穹集合通信库IHCCM，支持基于CPU和GPU的2种通信方式，以解决不同类型显卡间的通信问题。

（2）硬件加速技术的优化利用：异构硬件的加速能力是实现高性能计算的关键。然而，不同硬件平台的加速机制各不相同，如GPU的CUDA加速、FPGA的可编程逻辑加速以及CPU的多核并行处理等。这使得集合通信库必须针对每种硬件进行专门的优化，以充分利用这些加速技术，实现数据传输的高效性。

为了优化利用硬件加速技术，集合通信库需要实现硬件感知的算法和调度策略。一方面，通过硬件抽象层（HAL）来屏蔽底层硬件的具体实现细节，为上层算法提供统一的接口来访问硬件加速功能；另一方面，利用编译器技术，如即时编译（JIT）或提前编译（AOT），根据目标硬件

的特性生成优化的代码，以实现算法的高效执行。例如，智源的FlagScale通过异构流水线并行和异构数据并行2种模式，实现了高效的训练效率，接近同构训练的性能。

2.3　动态拓扑与容错机制的挑战

在智算场景中，随着云计算、边缘计算、物联网等技术的快速发展，计算节点的动态性日益显著。这些节点可能因任务需求、资源分配、能源管理或网络条件等多种因素而频繁加入或退出系统，从而形成一个高度动态、异构且复杂的计算环境。

（1）通信路径频繁变化：通信路径的频繁变化是动态拓扑环境中的一大挑战。由于节点的动态加入和退出，网络拓扑结构不断演变，原有的通信路径可能变得不再最优，甚至失效。例如，研究表明超过99%的GPU并不承载网络流量，而不到0.25%的GPU承载了模型并行和数据并行流量，这些流量类型占总传输数据的90%以上。这意味着当网络拓扑发生变化时，非关键路径上的通信可以被优化掉，但关键路径则需要重新评估以维持高效的数据传输。因此，集合通信库必须具备快速感知并适应这种拓扑变化的能力，能够动态地重新计算最优通信路径，确保高效、低时延的通信。

（2）通信故障与节点失效：动态拓扑变化不仅导致通信路径的频繁变动，还可能引发通信故障和节点失效，对集合通信库的容错机制提出极高的要求。为了确保数据的一致性和可靠性，集合通信库必须能够及时发现并处理这些故障，同时保证系统的持续运行。例如，阿里云提出的C4通信驱动加速方案，开发了一套容错机制，允许系统检测到节点失效后自动重新分配任务给其他健康的节点，从而保证了训练过程的连续性和数据的一致性，将错误引起的开销减少约30%，并将某些通信成本适中的运行时性能提高约15%。

2.4　通信开销与成本控制的挑战

（1）带宽消耗：在智算场景中，大规模节点集群的带宽分配与调度问题尤为复杂。随着节点间通信需求的增加，带宽资源变得愈发紧张，带宽竞争现象频发，严重影响了通信性能。特别是在AI模型训练过程中，大量的参数更新和数据传输需要占用大量的带宽资源，如何高效利用有限的带宽资源成为亟须解决的问题。例如，在Llama 3 000亿参数大模型的训练中，相比前一代模型，所需的数据传输量增加了7倍以上，这对网络带宽构成了巨大挑战。为了解决这一问题，业界采取了多种措施来优化带宽使用效率，包括采用新一代AI以太网技术和高性能网络设备，如华为AI Fabric实现了整网吞吐量高达98%，有效减少了带宽竞争现象。

（2）时延与功耗：通信时延的增加会直接影响AI模型训练的效率，而功耗的增加则会导致系统能耗的攀升，进而增加运营成本。在智算场景下，如何降低通信时延和功耗成为提高系统性能的关键。例如，NVIDIA推出的DGX GH200 AI超级计算机利用NVLink互连技术显著减少了通信时延；Google TPU通过低精度计算降低了功耗；Meta自研的MTIA芯片基于RISC-V架构，在保证性能的同时实现了更低的能量消耗。

2.5　安全与隐私保护的挑战

在智算场景下，数据的安全性和隐私保护同样至关重要。AI模型训练和推理过程中涉及大量敏感数据，如何确保这些数据在传输过程中的安全性和隐私性，成为集合通信库必须解决的重要问题。

数据传输过程中面临着诸多安全威胁，如网络攻击、数据泄露等。例如，美国社交媒体公司*X*未经告知或征得用户同意，擅自使用超过6 000万欧洲用户个人数据训练其大型语言模型Grok。若发生安全事故可能导致敏感数据的泄露和滥用，对用户的隐私造成极大威胁。同时，随着数据隐

私保护意识的增强，用户对数据隐私的需求也日益提高。因此，集合通信库必须采取有效的安全措施，确保数据在传输过程中的安全性和隐私性。

3 集合通信库的发展趋势

随着AI技术的迅猛发展，尤其是深度学习和大模型应用的日益普及，集合通信库作为分布式训练系统的核心组件，其重要性愈发显著。在面临不断增长的计算需求、日益复杂的训练场景以及多样化的硬件架构时，集合通信库必须持续进化，以适应这些新的挑战。

3.1 算法优化：深度定制与自适应调整

（1）深度定制通信算法：针对深度学习、大数据处理等特定应用场景，以及CPU集群、GPU加速卡、FPGA加速器等多样化的硬件架构，集合通信库需要开发一系列深度定制的通信算法。这些算法应充分考虑不同应用场景下的通信模式和数据特性，以及不同硬件平台的计算能力和通信能力。例如，在深度学习训练中，大规模参数同步是通信的主要瓶颈之一，为了解决这个问题，可以设计高效的分布式梯度聚合算法，如分层All Reduce算法，通过将大规模的梯度数据分成多个小块，并在多个节点之间进行分层聚合，从而有效减少通信量，加速模型的收敛速度。此外，还可以探索稀疏化通信算法，通过只传输重要的梯度信息来进一步减少通信开销。

（2）自适应算法优化：为了实现更高效的通信，集合通信库需要集成智能决策机制，根据当前的网络状况、任务特性和硬件资源动态选择最优的通信算法。这要求集合通信库内置一套完善的性能评估模型，能够实时监测并预测不同算法在不同条件下的表现。性能评估模型可以基于历史通信数据构建，利用机器学习技术对数据进行分析和建模，通过不断学习和更新模型，集合通信库可以准确预测不同算法在当前网络状况、任务特性和硬件资源下的性能表现，并实现算法的动态切换。这种自适应算法选择机制可以显著提高通信效率，降低通信时延，提升整体训练速度。

3.2 软硬件融合：深度整合与协同优化

（1）异构硬件协同：集合通信库需要开发能够跨CPU、GPU、FPGA等多种异构硬件平台高效运行的通信接口和协议。通过实现计算与通信的紧密耦合，可以充分利用不同硬件平台的计算能力，减少CPU的介入，提高整体效率。例如，在GPU加速卡上，可以利用直接内存访问（direct memory access，DMA）技术，实现数据在GPU之间的直接传输，绕过CPU，从而降低通信时延。通过深度整合多种硬件平台，集合通信库可以实现更高效的通信，提升整体训练速度。

（2）新型网络接口利用：随着高性能网络接口技术的发展，如RDMA、InfiniBand、Omni-Path等，集合通信库需要紧密结合这些接口的特性，设计低时延、高吞吐量的通信协议。例如，RDMA技术允许直接访问远程节点的内存，而无须通过操作系统内核进行数据传输。这可以显著减少数据传输的时延和开销，提高通信效率。集合通信库可以充分利用这些高性能网络接口的特性，设计优化的通信协议和算法，进一步提升通信性能。

3.3 安全增强：多层次防护与隐私保护

（1）端到端加密：为了确保数据在传输过程中的安全性，集合通信库需要在通信库层面实现端到端的数据加密。采用先进的加密算法（如AES、RSA）和密钥管理机制，可以保护数据免受窃听和篡改。同时，集合通信库还需要实现高效的密钥管理机制，确保密钥的安全性和可用性。

（2）隐私保护技术：除了端到端加密，集合通信库还需要引入差分隐私、联邦学习等隐私保护技术，以在保持数据隐私的同时进行模型训练或数据分析。差分隐私通过在数据传输或模型训练过程中添加适量噪声，保护个人隐私；联邦学

习允许多个节点在保持数据本地化的同时进行模型训练，实现模型的共享和更新而不暴露个人隐私数据。

3.4　技术融合：云原生与边缘场景支持

（1）云原生支持：随着云计算的普及和云原生技术的兴起，集合通信库需要更好地支持云原生环境，如Kubernetes集群。这要求集合通信库能够与云原生平台无缝集成，实现资源的弹性伸缩和高效利用。例如，集合通信库可以支持Kubernetes的容器编排和调度机制，实现通信资源的动态分配和管理。通过支持云原生环境，集合通信库可以更好地适应云计算场景下的分布式训练需求。

（2）边缘计算集成：随着物联网和边缘计算的兴起，集合通信库需要适应分布式、低时延的边缘计算场景。集合通信库需要提供轻量级、高效的通信解决方案，以满足边缘计算场景下的通信需求。例如，集合通信库可以优化通信协议和算法，降低通信开销和时延。同时，还可以利用边缘设备的计算能力和存储资源，实现数据的本地化处理和分析，进一步降低通信需求。通过集成边缘计算技术，集合通信库可以更好地支持分布式、低时延的智能计算应用。

4　结束语

在智算场景下，集合通信库是实现多节点数据高效交换的核心组件，对支撑大模型分布式训练至关重要。然而，随着算力规模的扩张和通信需求的激增，集合通信库正面临着多重挑战。一方面，算力规模与通信效率的双重压力使得通信时延和带宽瓶颈问题日益凸显，严重影响了模型训练的速度和效果；另一方面，异构硬件的兼容性、动态拓扑与容错机制，以及通信开销与成本控制等难题，也进一步加剧了集合通信库的复杂性和实现难度。此外，数据安全与隐私保护同样不容忽视，必须采取有效的措施确保数据在传输过程中的安全性和隐私性。

展望未来，集合通信库的发展趋势将聚焦于更高效的通信算法、更灵活的调度策略、更广泛的兼容性以及新兴技术的融合。通过深度定制通信算法和自适应算法选择，集合通信库能够根据实际场景和需求实现更高效的通信。同时，硬件融合与智能调度将进一步提升通信性能，降低时延和功耗。此外，集合通信库还需要增强安全性，实现多层次防护与隐私保护。随着云原生、量子通信和边缘计算等技术的不断发展，集合通信库将迎来更多的机遇和挑战，持续进化以适应智能计算时代的需求。

参考文献：

[1] OPENAI. Introducing ChatGPT[EB]. 2023.

[2] OPENAI. Introducing OpenAI o1[EB]. 2024.

[3] 郭亮, 王少鹏, 权伟, 等. 面向大模型的智算网络发展研究[J]. 电信科学, 2024, 40(6): 137-145.
GUO L, WANG S P, QUAN W, et al. Research on the development of intelligent computing network for large models[J]. Telecommunications Science, 2024, 40(6): 137-145.

[4] WEINGRAM A, LI Y K, QI H, et al. xCCL: a survey of industry-led collective communication libraries for deep learning[J]. Journal of Computer Science and Technology, 2023, 38(1): 166-195.

[5] KIM J, KIM H. Router microarchitecture and scalability of ring topology in on-chip networks[C]//Proceedings of the 2009 2nd International Workshop on Network on Chip Architectures. Piscataway: IEEE Press, 2009: 5-10.

[6] BOUKNIGHT W J, DENENBERG S A, MCINTYRE D E, et al. The illiac IV system[J]. Proceedings of the IEEE, 1972, 60(4): 369-388.

[7] 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.

[8] AL-DUBAI A Y, OULD-KHAOUA M. A new scalable broadcast algorithm for multiport meshes with minimum communication steps[J]. Microprocessors and Microsystems, 2003, 27(3): 101-113.

[9] THAKUR R, RABENSEIFNER R, GROPP W. Optimization of collective communication operations in MPICH[J]. International Journal of High Performance Computing Applications,

2005, 19(1): 49-66.

[10] Pjesivac-Grbovic J. Towards automatic and adaptive optimizations of MPI collective operations[J]. International Journal of High Performance Computing Applications, 2007, 15(3): 45-60.

[11] HUSE L P. Collective communication on dedicated clusters of workstations[M]//Recent Advances in Parallel Virtual Machine and Message Passing Interface. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999: 469-476.

[12] BRUCK J, HO C T, KIPNIS S, et al. Efficient algorithms for all-to-all communications in multi-port message-passing systems[C]//Proceedings of the Sixth Annual ACM Symposium on Parallel Algorithms and Architectures. New York: ACM Press, 1994: 298-309.

[13] BROWN T, MANN B, RYDER N, et al. Language models are few-shot learners[J]. Advances in neural information processing systems, 2020, 33: 1877-1901

[14] HUANG Y, CHENG Y, BAPNA A, et al. Gpipe: Efficient training of giant neural networks using pipeline parallelism[J]. Advances in Neural Information Processing Systems, 2019, 32.

[15] WANG G, QIN H, JACOBS S A, et al. ZeRO++: Extremely efficient collective communication for large model training[C]// The Twelfth International Conference on Learning Representations(ICLR), Vienna, Austria:ICLR,2024.

[16] RAJBHANDARI S, RASLEY J, RUWASE O, et al. ZeRO: memory optimizations toward training trillion parameter models[C]//Proceedings of the SC20: International Conference for High Performance Computing, Networking, Storage and Analysis. IEEE Press, 2020: 1-16.

[17] LIAN X R, HUANG Y J, LI Y C, et al. Asynchronous parallel stochastic gradient for nonconvex optimization[C]//Proceedings of the 29th International Conference on Neural Information Processing Systems - Volume 2. New York: ACM Press, 2015: 2737-2745.

[18] GRAHAM R L, SHIPMAN G M, BARRETT B W, et al. Open MPI: a high-performance, heterogeneous MPI[C]//Proceedings of the 2006 IEEE International Conference on Cluster Computing. Piscataway: IEEE Press, 2006: 1-9.

[作者简介]

高翔（1991− ），男，中国电信股份有限公司研究院工程师，主要研究方向为计算机系统结构、存算一体、异构计算、智算基础设置等。

董斌（1972− ），男，中国电信股份有限公司上海分公司正高级工程师，主要研究方向为人工智能与算力、5G业务平台等。

肖晴（1970− ）女，博士，中国电信股份有限公司上海分公司正高级工程师，主要研究方向为智算基础设施架构、公共智算服务平台、人工智能大模型应用等。

简晟（1999− ），男，中国电信股份有限公司研究院工程师，主要研究方向为人工智能、智算网络、算力网络等。

傅德基（1986− ），男，中国电信股份有限公司研究院工程师，主要研究方向为云计算领域的先进架构芯片、下一代虚拟化、技术长期演进创新机制等。