分布式深度学习系统网络通信优化技术

Optimization Techniques of Network Communication in Distributed Deep Learning Systems

董德尊 /DONG Dezun，欧阳硕 /OUYANG Shuo
（国防科技大学，中国 长沙 410073）
(National University of Defense Technology, Changsha 410073, China)

摘要：针对分布式深度学习系统网络通信的全协议栈定制优化问题，提出了一种分布式深度学习系统的网络通信优化技术的分类方法。从网络协议栈层次的角度，分析了通信流量调度和网络通信执行的关键技术；自顶向下地从算法层面和网络层面分别讨论了分布式深度学习通信瓶颈优化的几种基本技术途径，并展望其未来发展的机遇与挑战。

关键词：分布式深度学习系统；通信优化；全协议栈

Abstract: Aiming at optimizing the full protocol stack of the network communication in distributed deep learning systems (DDLS), a classification method of the network communication optimization techniques in DDLS is proposed. From the perspective of the entire network protocol stack, the key techniques of communication traffic scheduling and network implementation in DDLS are analyzed. Some basic techniques of bottleneck optimization of distributed deep learning communication from algorithm level and network level are discussed, and future research opportunities and challenges are identified.

Keywords: distributed deep learning systems; communication optimization; full protocol stack

DOI：10.12142/ZTETJ.202005002
网络出版地址：https://kns.cnki.net/kcms/detail/34.1228.TN.20200924.1853.004.html

网络出版日期：2020-09-25
收稿日期：2020-08-08

伴随人工智能研究的第 3 次浪潮，深度学习技术席卷了图像分类、语音识别、自动驾驶、内容推荐等众多应用领域。深度学习的普及促进了深度神经网络的发展，这些网络模型在各种各样的任务上都取得了良好的效果。以计算机视觉为例，一些经过精心设计的深度神经网络，例如在 ImageNet 数据集上训练的 GoogLeNet 和 ResNet-50，已经在图像分类任务上击败了人类。然而，训练高性能的网络模型常常需要花费大量时间，短则数小时，长则数天甚至数周。

为了减少训练时间，研究人员通常使用高性能硬件如图形处理器（GPU）和张量处理器（TPU）等加速深度神经网络模型的训练。同时，在多个节点上并行训练深度神经网络也是行之有效的加速方法。每个节点仅仅执行整体计算任务的一部分，所以这种分布式训练可以大幅缩短深度神经网络的训练时间，如图 1 所示。但是，由于网络模型训练过程的迭代性，不同的计算节点之间往往需要频繁地进行通信以交换大量的数据，这就导致节点间的通信成为分布式训练中的关键瓶颈，图 2 描述了这种情况。随着集群规模的扩大，节点间的通信开销会急剧地增加。这种现象极大地削弱了分布式训练所带来的优势，因为很大一部分时间都花在了网络数据传输上。当我们使用高性能硬件训练网络模型时，通信时间占整体训练时间的比例会进一步增加，因为这些高性能硬件只减少计算开销而没有降低通信开销。高昂的通信开销限制了分布式训练的可扩展性。为此，研究者们针对分布式训练过程中的通信行为，展了一系列的优化工作。

1 分布式深度学习系统的基本概念

当网络模型的训练任务被部署到

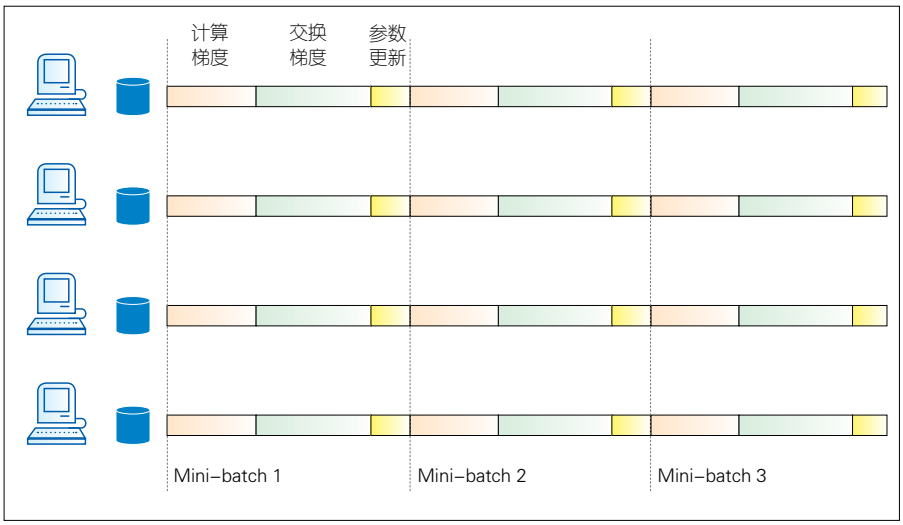
多个节点上时，我们需要考虑以下几个问题：并行化训练任务的哪一部分？如何组织计算资源？如何协调各个计算节点？

1.1 数据与模型并行

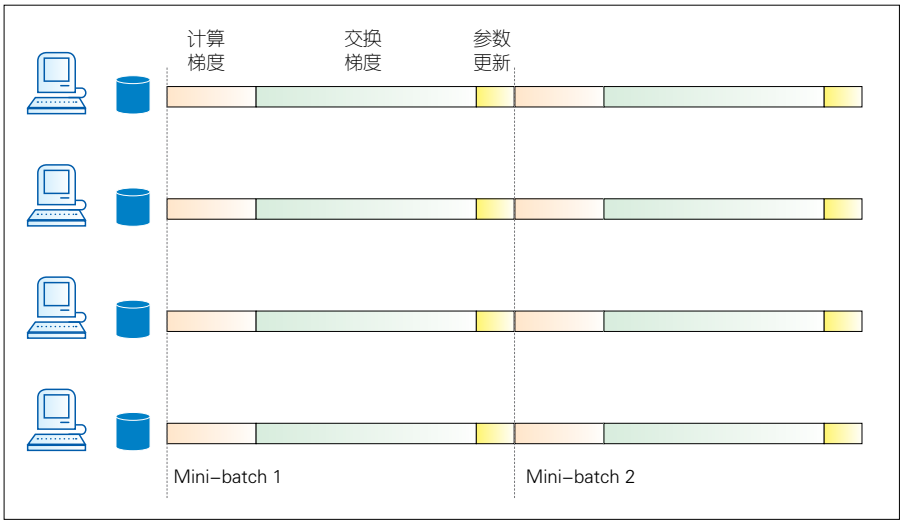
分布式深度学习系统面临的第一个问题是并行化训练任务的哪一部分。常见的两种并行模式为数据并行和模型并行，如图 3 所示。

在数据并行中，首先整个数据集被随机且均匀地分配到各个节点中，每个节点都在本地维护一个完整的模型副本。每个节点仅读取和处理唯一的数据子集，并在训练期间更新本地模型。然后，将这些本地模型参数与其他节点同步以计算全局参数。因此，这些全局参数利用网络分配到每个节点上，以便开始下一次迭代。

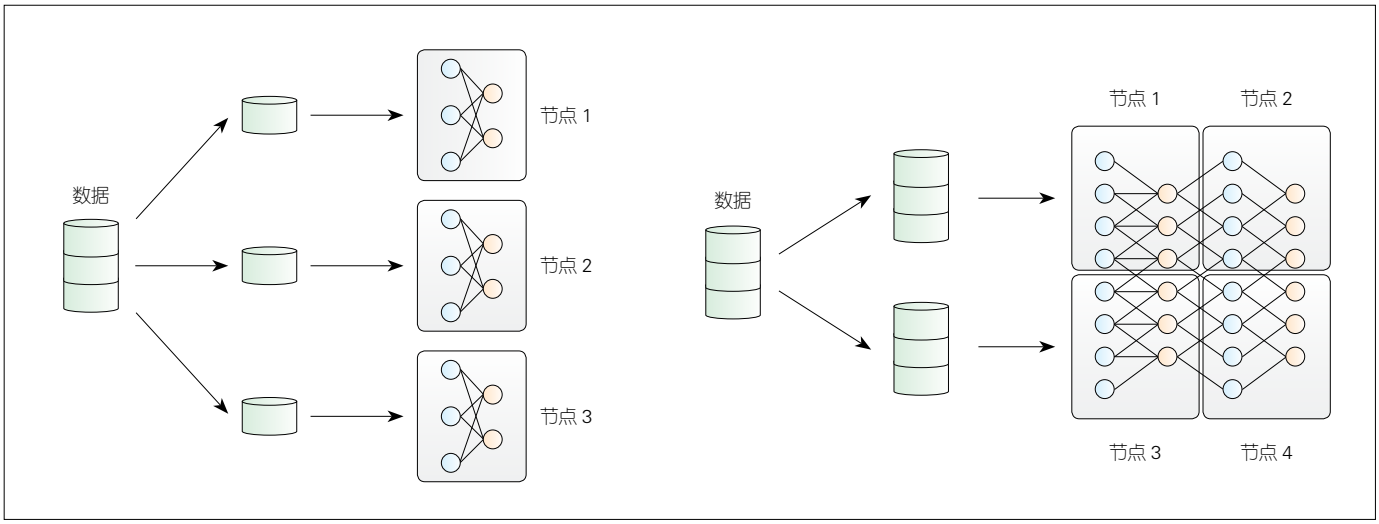
除了数据并行，模型并行是另一种方法。该方法将神经网络模型分割到不同的计算节点上，每个计算节点仅负责计算模型的某一部分，只有输入层所在的节点才负责读取数据。当模型太大而无法放到一台机器中时，则需要使用模型并行。本文主要关注常见的数据并行技术。



▲图 1 计算与通信相对均衡的分布式训练



▲图 2 通信开销占比过大的分布式训练



▲图 3 数据并行与模型并行

1.2 中心化与去中心化架构

分布式深度学习系统所面临的第二个问题是以何种结构组织计算节点。系统的架构会影响训练过程中数据的传输方式和模型更新方式，进而影响训练时间。常见的架构包括中心化以及去中心化架构。

参数服务器架构是分布式深度学习中最常见的中心化架构。参数服务器架构通常包括若干服务器和若干工作节点。其中服务器上存放着全局共享的模型参数。如果有多个服务器节点，那么模型的权重参数会被拆分到每台机器上。每个工作节点都存储一个模型副本（使用数据并行）或存储模型的某一部分（使用模型并行）。工作节点通过推送 / 拉取操作与服务器进行通信，而任何工作节点之间都不会产生通信行为。每次迭代中，每个工作节点首先读取数据，然后基于这些数据计算本地模型的梯度。随后，节点将其本地梯度推送到参数服务器。服务器接收所有工作节点发送的梯度后，首先聚合这些梯度，然后更新全局模型，最后工作节点再从服务器拉取最新的模型权重，并使用它进行下一次的迭代。

点对点架构是一种去中心化的架构，其中每个节点在模型训练中都扮演相同的角色。与参数服务器类似，点对点中的每个节点都拥有完整的模型副本，并首先基于小批量数据计算本地梯度。对等体系结构使用归约和广播等集合通信操作，而不是参数服务器体系结构中的推送 / 拉取操作。经过梯度计算后，每个对等方首先从其他对等方接收梯度，然后对这些梯度求平均，这就是归约步。然后，对等方将其本地梯度广播给所有其他对等方。节点从其他对等节点收到所有梯度后，立即更新其本地模型，然后执行下一个迭代。

1.3 同步与异步更新

将训练任务并行化到多个节点上时，如何协调这些节点是一个大问题。同步更新和异步更新是当前的两种主流方法。

使用同步更新时，两次迭代之间存在全局屏障。网络中的节点首先计算局部梯度，然后将梯度与其他节点同步。由于各种因素，每个节点都有不同的完成时间。在一次迭代中，由于存在全局屏障，计算较快的节点不得不等待计算较慢的节点。全局屏障的存在导致了“掉队者”的问题，较快的节点需要大量的等待时间。

为了克服此限制，研究人员提出了异步更新方案。此方案中，两次迭代之间没有全局屏障，每个节点在从其他节点接收到参数之后（无论这些参数是旧参数还是新参数），都立即开始下一次迭代。每个节点之间没有同步屏障，因此某些节点在更新全局模型参数时可能会使用过时的模型参数来计算梯度。数据的陈旧性将偏差和方差导入局部梯度和全局模型参数中，这会导致模型收敛缓慢且没有收敛性保证。为了应对掉队者和数据陈旧性的问题，延迟异步更新被提出。该方案是同步更新和异步更新的结合，并对数据的陈旧性做了限制。

2 分布式深度学习系统网络通信优化技术

为了有效地在多节点上训练深度神经网络，降低训练过程中的通信开销，业界已经提出了多种通信优化策略。由于分布式深度学习通信优化涉及深度神经网络、异构并行计算、分布式系统、计算机网络等众多技术，从芯片级到系统级有许多涉及网络通信优化相关的技术。

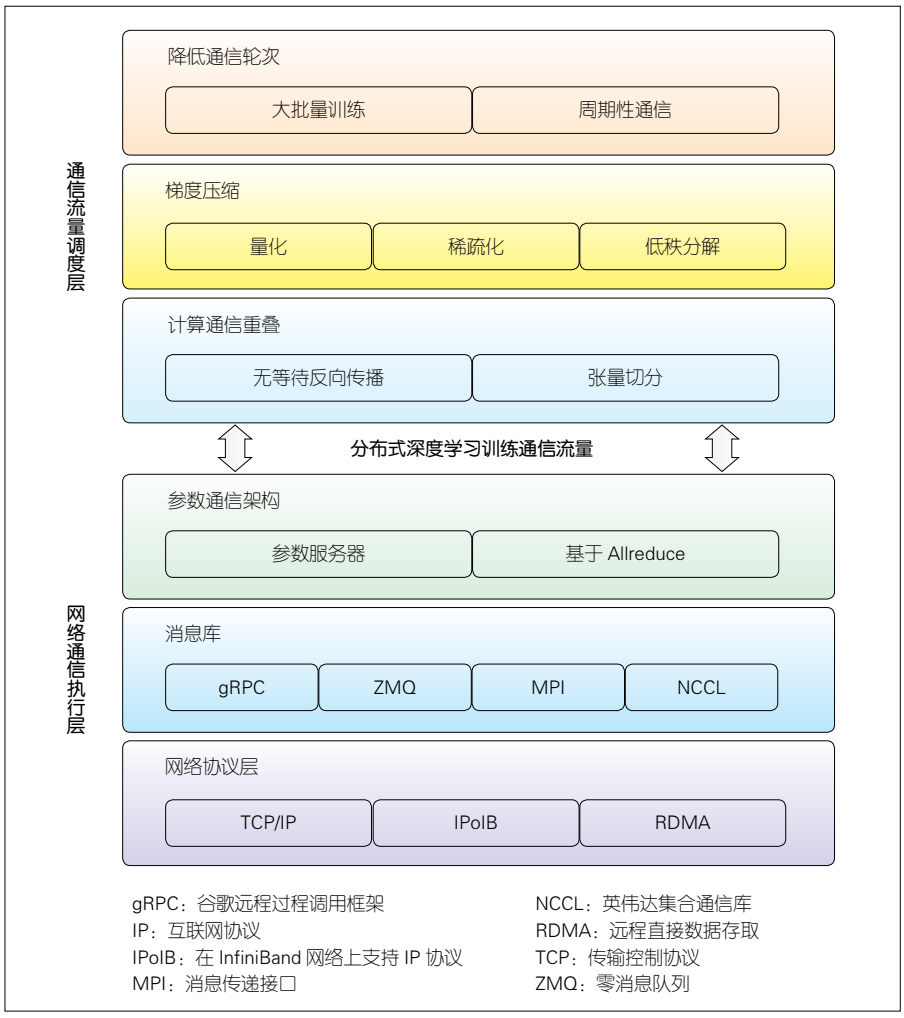
本文尝试从网络协议栈层次的角度，从通信流量调度层和网络通信执行层的角度，对分布式深度学习系统的网络通信优化技术进行初步分类讨论，如图 4 所示。从通信流量调度层的角度来看，可以通过降低通信发生的频次，来降低通信数据量（梯度压缩）以及计算通信重叠等技术优化分布式训练的通信过程。在网络通信执行层面，不同的参数通信架构、不同的消息传递库以及不同的网络协议都会对通信产生影响。网络与通信领域的研究人员，对网络通信执行层面的优化技术往往更为熟悉，实际上网络底层主要采用的还是通用的网络技术。为了追求高性能，分布式深度学习系统的网络通信优化必然是对通信全协议栈的定制优化和协同设计。从这个角度来看，定制优化和协同设计需要深入分析分布式深度学习训练系统的通信需求，充分利用底层网络技术的特点，从而对分布式深度学习训练系统的通信流量进行高效调度与优化。

2.1 降低通信发生的频次

训练深度神经网络时，整个训练过程通常需要进行多次 epoch 和迭代。降低通信开销的一种直接方法是减少通信轮次，而通信轮次与批量大小和通信周期有关。较大的批量和较长的通信周期都可以减少数据交换次数。

2.1.1 使用大批量进行训练

批量大小控制每次迭代时读取的数据量。在基于数据并行的分布式训练中，批量通常指的是全局批量，即所有节点的本地批量的总和。对于并行化训练，节点通常在每次迭代结束时交换数据，因此，当训练数据集的大小固定时，增加批量会减少迭代次数，从而减少通信轮次。梯度和参数的形状及大小仅取决于神经网络模型本身，因此单次迭代传递的消息大小总是保持不变，更改批量不会改变每



▲图 4 通信优化层次

次传输的消息总量[1]。大批量带来的好处是更少的通信轮次。由于具有大批量训练的优势，分布式深度学习系统网络通信优化技术最近的工作将批量增加到 8 k、32 k 甚至增加到 64 k 样本（1 k 表示 1 024 个样本）。然而，当固定 epoch 时，与小批量相比，实际上直接使用具有大批量的随机梯度下降（SGD）通常会带来泛化能力的下降。由于梯度估计存在方差，小批量 SGD 始终收敛到平坦的最小值，而大批量 SGD 趋向于收敛到尖锐的最小值——尖锐的最小值导致较差的泛化能力。通常，神经网络模型的泛化能力相较于训练速度来说更为重要。因此，我们需要保证在运行相同的 epoch 下，使用大批量训练的模型具有与小批量训练的模型相近的泛化能力。

2.1.2 周期性通信

通信轮次不仅与批量的大小有关，还与通信周期相关。前面提到，通信操作常常发生在每次迭代的末尾。因此，我们可以控制训练进程，每隔几次迭代进行通信，进而降低通信操作发生的次数。实际上，最常用的方法是本地 SGD[2]。为了降低多次通信带来的巨大开销，本地 SGD 让每个工作节点先在本地进行若干次迭代，然后再与参数服务器同步梯度和全局参数，如图 5 所示。实验证明，只要周期设置合适，本地 SGD 可以减少训练时间的通信开销并且保证收敛性。

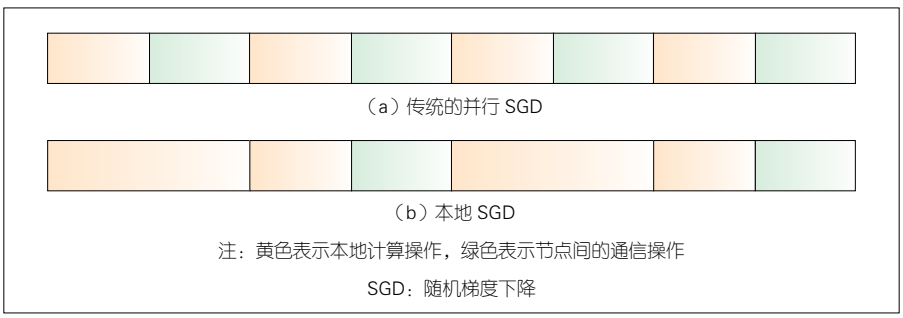
2.2 降低通信过程中传输的数据量

在传统的深度神经网络分布式训练过程中，计算节点之间交换梯度和模型参数，以进行模型聚合和本地计算。当梯度和参数量很大时，由于交换大量 32 位浮点变量而导致的通信瓶颈，削弱了并行化带来的优势。

解决此问题的一个直接想法是压缩传输的梯度，最常用的两种方法是梯度量化和梯度稀疏化。前一种方法使用低精度数字（即 8 位、4 位甚至 1 位）替换 32 位浮点数，以此来减少传输的梯度量；而后一种方法则选择了梯度向量中的一些重要元素来更新模型参数，以避免不必要的传输开销。

2.2.1 梯度量化

梯度和模型参数中的每个元素都存储在电气和电子工程师协会标准（IEEE 754）的单精度浮点变量中。交换大量的 32 位全精度变量经常占用大量网络带宽。为了缓解这种通信瓶



▲图 5 传统并行 SGD 与本地 SGD 的对比

颈，研究人员尝试使用低精度浮点数来表示训练过程中传输的梯度。

梯度量化的一般性描述为：首先梯度在发送之前通过量化函数量化为低精度值，然后在模型训练期间，每个节点之间传输这些低精度值；接收节点通过反量化函数从量化值重构原始梯度，再聚合这些重构后的梯度向量并更新模型参数。根据所使用的量化函数的性质，量化操作可以分为确定性量化和随机性量化。

确定性量化将梯度元素值映射到某些固定的值。1 bit SGD[3] 是确定型量化的典型例子，它根据梯度元素的取值范围，将所有的梯度值量化到 0 和 1 这两个数字。接收端在收到量化的 01 序列后，会把其中的 0 解码成 −1，把 1 解码成 +1，再进行下一步的训练。通常来说，因为确定性量化的固定量化形式会有较多的信息丢失，随机性量化应运而生，这种方法使得量化后的梯度元素值仍然服从某一概率分布。该方法引入了额外的随机性，因此量化后的梯度一定是原始梯度的无偏估计，并且必须具有方差约束以确保具有像传统 SGD 一样的收敛性。典型的随机性量化方法如量化 SGD（QSGD）[4] 和 TernGrad[5] 等。

2.2.2 梯度稀疏化

由于我们需要至少 1 位来表示梯度向量中的每个元素，因此梯度量化方法最多只能将数据量压缩 32 倍。梯度稀疏化则没有上述压缩率的限制，它只关注更新过程中梯度向量和模型参数中的一些重要值，即那些值远大于零的数字。根据选择的元素数量，稀疏化方法可以达到非常高的数据压缩率。稀疏化方法的核心是如何从梯度向量中选择有效值，即如何将稠密更新转换为稀疏更新。常见的稀疏化方法是 Top-K，它保留了梯度向量中前 K 个较大的绝对值，而将其他值设置为零。比如，基于 Top-K 的 Gradient Dropping 算法 [6] 在手写字符识别和机器翻译等任务中取得了良好的加速效果。关于梯度压缩，如何平衡模型精度与数据压缩比率是一个严峻的挑战。误差补偿已被证明是一种行之有效且兼容大部分压缩算法的技术 [7]。此外，还有研究者使用了动量屏蔽和梯度裁剪等技术保证模型的精度 [8]。另外，研究者还应该关注如何降低梯度压缩技术的计算开销，以进一步加速模型训练。

2.3 计算与通信重叠

现有的深度学习框架后端引擎以先进先出（FIFO）的顺序执行操作。因此，按其生成顺序发送梯度，最后一层（输出层）的梯度先被发送，然后处理中间层，最后处理输入层。要完成当前迭代并尽早地开始下一次迭代，我们需要减少计算与第一层通信之间的延迟。

Poseidon 系统提供了一种基于分布式训练的固有特性的无等待反向传播调度算法 [9]：一旦反向传播计算出某一层的梯度，后端引擎就开始传输该层的梯度数据。但是，不同层的参数量可能并不相同，因此具有不同的计算和通信时间。这就意味着 Poseidon 不一定会比某些特定网络模型上的 FIFO 调度表现更好。

P3[10]（Priority-base Parameter Propagation）通过基于优先级的调度扩展了无等待反向传播。在 P3 中，靠近输入层的梯度向量具有较高的优先级，而靠近输出层的梯度具有较低的优先级。在训练阶段，无论高优先级的梯度向量在何时生成，都将优先对其进行处理。这样就保证了其他节点能够尽早地接收到靠近输入层的梯度，从而能够尽早开始下一次迭代。此外，P3 使用张量分割技术将各层的参数向量分解为适当的小块，并根据其所在的层为每个切片分配优先级，以实现更细粒度的流水线化。高效的计算通信重叠率可以显著地加速模型训练过程。但是，当前大部分调度算法都是启发式的，这就意味着调度算法并不是最优解。因此，寻找高效的计算、通信调度算法至关重要。目前，已经有研究者将贝叶斯优化算法 [11] 和强化学习 [12] 应用在算子调度上。

单步延迟 SGD（OD-SGD）[13] 算法打破了下一次迭代计算对上一次迭代中通信过程的依赖，通过实现计算过程与通信过程的高度重叠来提升分布式训练性能。该算法结合了同步 SGD 和异步 SGD 两种更新算法的优势，在保证训练精度的情况下提高分布式训练的速度。图 6 展示了在同步 SGD 算法训练模式和在 OD-SGD 算法训练模式下的性能对比。假设单次迭代的计算开销和通信开销均为 3 个时间单位，则在同步 SGD 算法训练模式下，单次迭代训练的时间开销为 5 个时间单位；而在 OD-SGD 算法训练模式下，单次迭代的时间开销为 3 个时间单位，原来额外的 2 个时间单位的通信开销被隐藏。

2.4 参数通信架构

在去中心化架构下，分布式深度学习的训练过程符合全局规约操作的语义：每个节点独立计算局部梯度，然后通过全局规约运算来计算梯度总和并将其发送给所有对等节点。显然，上述操作可以看作一个 Allreduce 运算。因此，整个网络通信过程的性能也就取决于 Allreduce 的性能。

在高性能计算领域，关于 Allreduce 算法的研究已经非常充分了。Allreduce 是近两年才在深度学习训练系统中得到广泛应用的，其中很有代

表性的是 Ring-Allreduce，由百度于2017年引入应用到主流的深度学习框架中。Ring-Allreduce由两个阶段组成：Reduce-Scatter和Allgather。当使用p个计算节点时，每个阶段都包括$p-1$个通信步骤。每个节点都维护其本地梯度，本地梯度被平均分为p块。在Reduce-Scatter阶段，每个节点发送和接收张量的不同块。在$p-1$个步骤之后，每个节点都具有了一部分块的全局结果。在Allgather阶段，每个节点发送自己维护的部分全局结果，并从其他节点接收其他部分全局结果。同样是$p-1$个步骤之后，每个节点都拥有完整的全局结果，如图7所示。因此，Ring-Allreduce总共需要$2\times(p-1)$个通信步。

此外，关于参数服务器架构及通信特征的详细分析，请参考文献[14]。

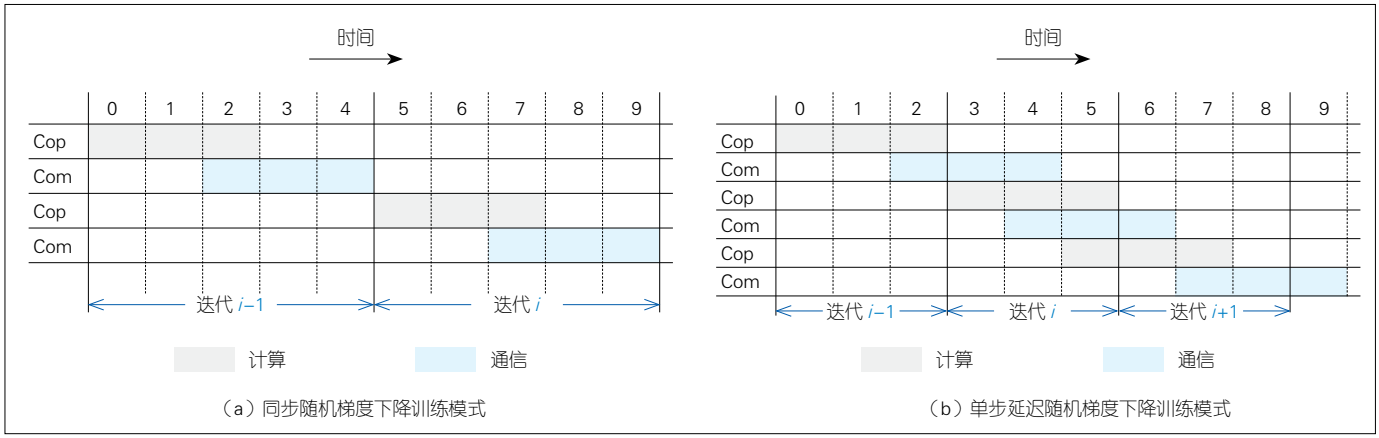
2.5 网络消息库

在去中心化架构下，目前关于Allreduce的实现非常多，包括NVIDIA的集合通信库、Facabook的Gloo以及百度的Allreduce等。以NCCL为例，它针对NVIDIA的GPU，实现了单机多卡以及多机多卡之间的高效GPU通信。在当前主流的跨平台统一分布式训练框架Horovod[15]中，NCCL、Gloo以及MPI都已经得到了支持。

对于参数服务器架构来说，由于其固有的“少对多”的通信模式，因此在常见的实现中，底层通信模块总是依赖于点对点的消息库，例如TensorFlow中的gRPC，以及MXNet中默认使用的ZMQ消息库。尽管MPI中也实现了低延迟的点对点通信，但是在参数服务器架构下并不能发挥出MPI在集合通信方面的优势。

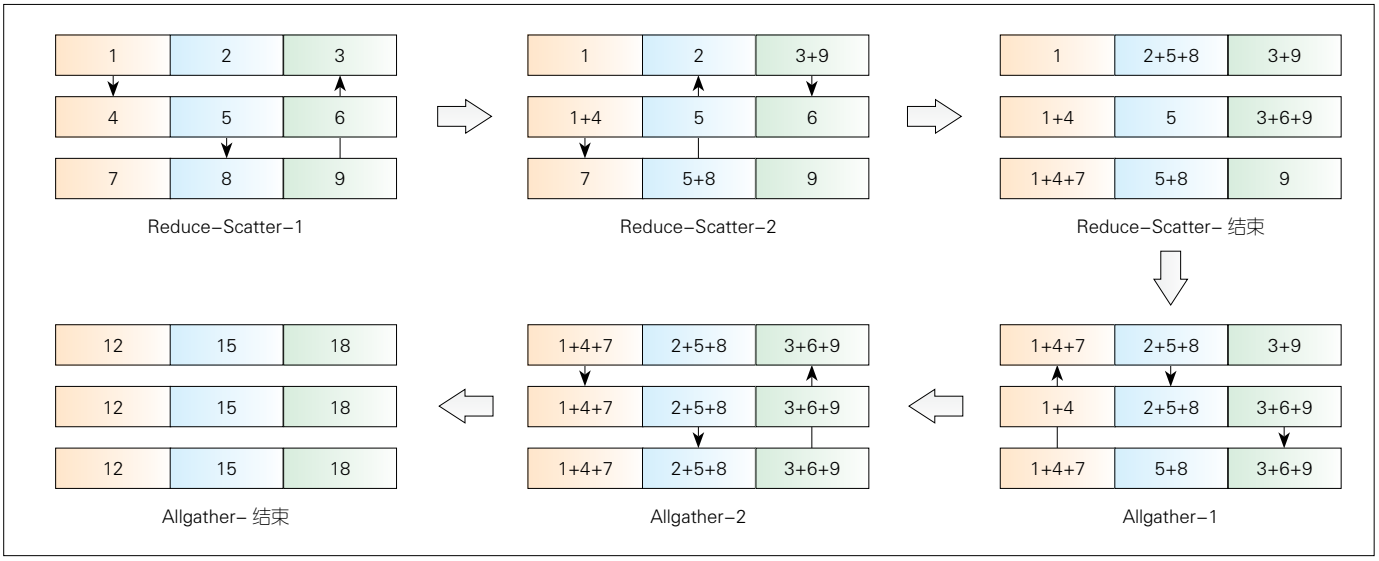
2.6 网络协议优化

早期的分布式训练框架的通信模块实现通常基于传输控制协议（TCP）/互联网协议（IP），需要先将参数数据复制到内核态的网络协议栈中，再通过网



（a）同步随机梯度下降训练模式　　（b）单步延迟随机梯度下降训练模式

▲图6 传统同步随机梯度下降与单步延迟随机梯度下降的对比



▲图7 Ring-Allreduce过程

络接口发出去，这些复制操作增加了分布式训练的通信延迟。远程直接内存访问（RDMA）允许用户态进程直接读取和写入远端进程的地址空间，是传统高性能计算系统中常用的高带宽低延迟的通信技术。在当前主流的分布式训练框架中，RDMA 原语替换了原始的基于套接字和 TCP 的接口。实验表明，使用 RDMA 替换传统的 TCP/IP 协议可以大幅降低分布式训练的同步开销，提升训练速度，扩大训练规模。

3 结束语

通信开销是扩展大规模深度学习的障碍，训练过程中高昂的通信代价令人难以接受。本文主要从算法和网络两个层面介绍了分布式深度学习中的通信优化策略，其中大部分优化策略都是正交的，它们可以进行组合，以进一步降低分布式深度学习训练过程中的通信开销，加速网络模型训练。关于神经网络模型训练，还有一个重要的研究方向是关于性能模型以及测量工具的探索与实现。性能模型能够帮助我们在理论上分析各类开销，而测量工具则允许我们找到分布式训练中的各类瓶颈。尽管 TensorFlow[16] 与 MXNet 都提供了相关工具，但是我们仍然需要更高级的网络分析工具如 Horovod Timeline[15] 以及 SketchDLC[14] 等。

致谢

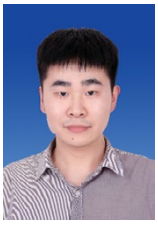
本文的部分研究成果和内容撰写得到国防科技大学徐叶茂博士生的帮助，谨致谢意！

参考文献

[1] YOU Y, ZHANG Z, HSIEH C, et al. Fast deep neural network training on distributed systems and cloud TPUs [J]. IEEE transactions on parallel and distributed systems, 2019, 30(11): 2449–2462. DOI:10.1109/TPDS.2019.2913833
[2] STICH S U. Local SGD converges fast and communicates little [C]//Proceedings of International Conference on Learning Representations (ICLR). Vienna, Austria: ICLR, 2019
[3] SEIDE F, FU H, DROPPO J, et al. 1–bit stochastic gradient descent and its application to data–parallel distributed training of speech DNNs [C]//Proceedings of Fifteenth Annual Conference of the International Speech Communication Association. Shanghai, China: Interspeech, 2014: 1058–1062
[4] ALISTARH D, GRUBIC D, LI J, et al. QSGD: Communication–efficient SGD via gradient quantization and encoding [C]//Proceedings of the 31st International Conference on Neural Information Processing Systems. Red hook, NY, United States, 2017: 1707–1718. DOI:10.5555/3294771.3294934
[5] WEN W, XU C, YAN F, et al. Terngrad: ternary gradients to reduce communication in distributed deep learning [C]//Proceedings of the 31st International Conference on Neural Information Processing Systems. Red hook, NY, United States, 2017: 1508–1518. DOI:10.5555/3294771.3294915
[6] AJI A F, HEAFIELD K. Sparse communication for distributed gradient descent [C]// Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. Stroudsburg, PA, USA: Association for Computational Linguistics, 2017: 440–445. DOI:10.18653/v1/d17–1045
[7] TANG H, YU C, LIAN X, et al. DoubleSqueeze: parallel stochastic gradient descent with double–pass error–compensated compression [C]//Proceedings of International Conference on Machine Learning. CA, USA: PMLR, 2019: 6155–6165
[8] LIN Y, HAN S, MAO H, et al. Deep gradient compression: reducing the communication bandwidth for distributed training [C]//Proceedings of International Conference on Learning Representations (ICLR). Vancouver, BC, Canada: ICLR, 2018
[9] ZHANG H, ZHENG Z, XU S, et al. Poseidon: an efficient communication architecture for distributed deep learning on GPU clusters [C]// Proceedings of the 2017 USENIX Conference on USENIX Annual Technical Conference (ATC). Santa Clara, CA, USA: ATC, 2017: 181–193
[10] JAYARAJAN A, WEI J, GIBSON G, et al. Priority–based parameter propagation for distributed DNN training [C]//Proceedings of 2nd Conference on Systems and Machine Learning (SysML). Austin, TX, USA: ACM, 2019
[11] PENG Y, ZHU Y, CHEN Y, et al. A generic communication scheduler for distributed DNN training acceleration [C]//Proceedings of the 27th ACM Symposium on Operating Systems Principles. USA: ACM, 2019: 16–29. DOI:10.1145/3341301.3359642
[12] HASHEMI S H, JYOTHI S A, CAMPBELL R H. TicTac: Accelerating distributed deep learning with communication scheduling [C]//Proceedings of 2nd Conference on Systems and Machine Learning (SysML). Stanford, CA, USA: SysML, 2019
[13] XU Y, DONG D, ZHAO Y, et al. OD–SGD: One–step delay stochastic gradient descent for distributed training [J]. ACM transactions on architecture and code optimization, preprint. DOI: 10.1145/3417607
[14] XU Y, DONG D, XU W, et al. SketchDLC: a sketch on distributed deep learning communication via trace Capturing [J]. ACM transactions on architecture and code optimization, 2019, 16(2): 1–26. DOI: 10.1145/3312570
[15] SERGEEV A, DEL BALSO M. Horovod: fast and easy distributed deep learning in tensorflow [EB/OL]. (2018–02–15) [2020–09–11]. https://arxiv.org/abs/1802.05799
[16] ABADI M, BARHAM P, CHEN J, et al. Tensorflow: A system for large–scale machine learning [C]//12th USENIX Symposium on Operating Systems Design and Implementation (OSDI). Savannah, GA, USA: OSDI, 2016: 265–283

作 者 简 介

董德尊，国防科技大学计算机学院研究员，天河互连系统副主任设计师；主要研究领域为高性能互连网络、数据中心网络、智能计算系统等；获多项省部级科研奖励；发表论文 50 余篇。



欧阳硕，国防科技大学计算机学院在读硕士研究生；主要研究方向为分布式深度学习系统。