面向大模型场景的异构算力分布式并行训练方法

黄蕾[1]，王升[1]，班有容[1]，张昊[1]，张晓光[1]，狄新凯[1]，许思[2]，黄子潇[2]
（1.中国移动研究院，北京 100053；2.上海无问芯穹智能科技有限公司，上海 200232）

摘　要：当前，不同类型AI加速器之间存在“资源墙”，难以聚合异构智算资源集群成池，以支持更大规模模型的训练孵化。基于此，设计了异构混合并行训练总体技术架构，并针对计算任务拆解及优化、分布式策略性能预测与生成、异构芯片间统一通信库三大关键技术方向提出了解决方案。其中，异构混合训练非均匀计算任务切分算法通过计算负载均衡依据算力大小和计算特性为多厂商智算集群分配计算任务；分布式策略性能预测及生成工具通过构建策略搜索空间模拟计算不同并行策略性能数据，输出最优非均匀并行切分策略；多厂商互识的统一异构通信库通过统一通信组件、异构通信组件、设备适配器实现通信拓扑管理、通信域管理等，解决异构AI加速器间数据无法互通问题。研发了基于异构混训技术的原型系统，在Nvidia GPU、天数智芯、壁仞组成的异构混合集群上进行了实验。实验结果表明，异构芯片集群的交叉混合训练加速比均超过90%，混合训练技术方案可行，且能够有效优化集群训练性能指标。

关键词：大模型；分布式训练；异构混合训练；深度学习框架；集合通信
中图分类号：TP391
文献标志码：A
DOI：10.11959/j.issn.1000–436x.2025167

Distributed parallel training technology for large-scale model with heterogeneous computing resources

HUANG Lei[1], WANG Sheng[1], BAN Yourong[1], ZHANG Hao[1], ZHANG Xiaoguang[1], DI Xinkai[1], XU Si[2], HUANG Zixiao[2]
1. China Mobile Research Institute, Beijing 100053, China
2. Infinigence AI, Shanghai 200232, China

Abstract: Currently, the “resource wall” between different AI accelerators makes it difficult to build one heterogeneous resource pool for large-scale models training. Based on this, an innovative heterogeneous distributed parallel training technology architecture was proposed, and innovative solutions had been introduced for three key technical directions: inhomogeneous task distribution for heterogeneous distributed training, heterogeneous distributed training performance prediction, and unified heterogeneous communication library. In which, inhomogeneous task distribution method was used to allocate computing tasks to multi-vendor AI accelerator clusters according to their computing power and characteristics through computing load balancing. The heterogeneous distributed training performance prediction technology constructed a strategy search space to simulate and compute performance metrics across various parallel strategies, ultimately output the optimal configuration for training framework. Unified heterogeneous communication library enabled topology management and, communication domain management through its unified communication components, heterogeneous communication modules, and device adapters, which effectively addressing the data interoperability bottleneck between heterogeneous intelligent GPUs. Experimental results show that the training acceleration ratio of the proposed method in Nvidia GPU+BI-V150 and Nvidia GPU+BR 106B is over 90%, which means the hybrid training technology scheme is feasible and can effectively optimize the cluster training performance index.

Keywords: large-scale model, distributed training, heterogeneous hybrid training, deep learning framework, collective communication

收稿日期：2025–05–30；修回日期：2025–09–19
通信作者：王升，wangshengyjy@chinamobile.com
基金项目：国家自然科学基金资助项目（No.U24B6012）
Foundation Item：The National Natural Science Foundation of China (No.U24B6012)

0 引言

以通用预训练聊天大模型（ChatGPT）[1]、DeepSeek[2]、大语言模型元AI（LLaMA）[3]等为代表的大模型技术正持续推动社会变革，当前流行的大模型具有数千亿甚至上万亿参数规模，使用单一类型计算节点训练过程耗时巨大，较难满足训练任务动态扩容需求，需要充分整合可调用的算力资源进行分布式并行加速。当前数据中心内可用的算力资源类型多样，既有不同厂商的AI加速器，也有同厂商不同代际的AI加速器，这些芯片在计算架构、软件栈、互联方式等方面存在着较大差异，异构AI加速器“资源墙”的存在限制了多厂商、多类型资源的灵活协同，数据中心内多样性算力资源难以形成训练“合力”。

为充分利用各类型算力资源，构建智算融通生态，本文针对跨架构的混合并行训练技术进行了深入的研究探索。

1)提出了异构混合训练非均匀计算任务切分算法，解决了异构混合场景下不同AI加速器子集群的负载均衡问题。

2)提出了异构非均匀切分性能预测技术，通过异构智算集群性能分析预测模型和工具，实现最优分布式策略的自动化、智能化生成。

3)定义了一套面向多厂商互识的通信机制，解决了异构混合场景下不同架构AI加速器之间的任务和数据分发协同问题。

4)构建了异构混合算力的分布式训练原型系统，并在3种不同芯片组成的混合集群上进行了实验验证。实验结果表明，该技术方法能够有效优化混合算力集群的训练性能指标。

1 背景介绍

大模型训练对于算力有着极高的需求，目前大模型参数量已达数千亿、上万亿规模，且数据量、参数量上升趋势仍未停止，因此在数据中心构建一个大规模资源池显得愈发重要。

当前针对特定模型训练任务构建单一类型芯片万卡集群已有相关探索，然而面向特定业务构建大规模单一类型AI加速器集群需要大量人力、物力、资本等资源投入，伴随大模型迭代优化需要，AI加速器资源池应按需动态扩容，不可避免地存在存量AI加速器与新购置AI加速器跨代际混池训练的需求。同时，智算资源池在建设过程中每年规划的厂商类别存在差异，面向大规模参数量模型训练场景需求，充分利用数据中心已有各厂商、各代际的芯片，整合形成模型训练混合资源池成为提高数据中心各类AI加速器利用率的关键问题。另外，在构建超万卡、十万卡集群过程中，容易出现单厂商绑定问题，进而导致技术栈封闭及供应链风险，因此，构建异构混池集群形成混合训练能力是解决该问题的关键。

当前不同类型的芯片之间存在“资源墙”问题，即由于各类芯片之间存在架构设计、计算能力、数据支持类型、通信机制等差异，难以将不同类型的计算资源组成一个大的混合算力资源池共同支撑大模型的训练任务。因此，为了突破“资源墙”限制，使不同AI加速器之间形成“合力”支撑更大规模模型训练，需要从异构资源合理负载均衡角度出发，从分布式并行方式、训练数据协同方式等多个方面进行详细研究。

传统基于同构集群的大模型分布式训练通常会使用多种不同的并行策略，常见有数据并行（DP, data parallelism）[4]、张量并行（TP,tensor parallelism）[5]、流水线并行（PP, pipeline parallelism）[6]、零冗余优化器（ZeRO, zero redundancy optimizer）[7]等，实际场景中，往往会在分布式加速框架如Megatron、Deepspeed中组合使用上述并行策略，从而达到最佳的并行训练效果。

此外，在通信机制方面，传统面向同构AI加速器集群的分布式训练已经有了一套相对成熟的集合通信机制，包括拓扑感知、通信管理及多种集合通信算法等。例如，英伟达集合通信库（NCCL，NVIDIA collective communications library）是一个适用于Nvidia芯片的集合通信库，并且在底层面向Nvidia的硬件架构做了大量的针对性优化。

然而，这些并行策略和通信机制的设计局限于同架构算力资源，未考虑异构混合训练场景，要实现同一任务在异构资源上的联合训练，当前面临来自诸多方面的挑战，具体如下。

1) 如何对异构混合算力集群进行非均匀的计算任务分布式并行拆解是一个挑战。传统的分布式并行训练都是面向单一类型智算集群，因此其对于计算任务的拆解也较为简单，只需要根据集群和芯片数量对计算任务进行均匀的拆解和分配。然而对于异构混合训练的场景，可能存在不同芯

片之间的浮点运算速度不同、支持数据类型不同、显存不同等问题，在同一训练任务下，并行训练组在数据协同过程中，算力较强设备需等待算力较弱设备计算完成才能够进行数据传输同步，算力较弱设备成为计算短板，影响集群并行计算性能。同时，不同芯片计算逻辑存在差异，精度上有所区别，容易导致模型损失无法收敛等一系列复杂问题。因此，对计算任务进行非均匀分布式并行拆解十分关键。

2) 在异构混合算力场景下，如何为训练任务自动化地给出最优的组合并行策略是一个挑战。传统的分布式并行训练中，数据并行、张量并行、流水线并行等策略参数配置往往依赖于有经验的模型训练专家人为设定，这主要是因为同构集群场景下，芯片数量、设备数量、芯片显存大小等影响策略参数的变量相对可控，有经验的专家可以很快估算出最优策略的若干备选，并进行较少的单步训练即可判断出最优策略。当场景变为异构算力混池训练时，由于变量激增、场景复杂，通过人工方式评估最优策略变得几乎难以实现，需要通过预测算法去求解这一 NP-hard 问题。通过性能预测的方式自动化得出最优的组合并行策略变得格外重要。

3) 不同类型 AI 加速器之间计算任务的数据协同也面临着巨大挑战。不同厂商的芯片之间存在计算架构、互联方式等多重维度的区别，而在分布式并行训练过程中，芯片之间需要进行参数同步等数据通信行为，由于异构混训场景下不同类型芯片在互联方式、通信库、通信协议等层面均存在较大差异，从而带来数据传输不互通、通信带宽及时延有差异等问题，各类芯片无法互联互通，当前技术无法实现异构资源混池分布式并行训练数据传输协同。因此，需要综合考虑异构混训并行计算策略，设计一套面向多厂商的数据传输协同机制，实现异构混合分布式并行训练参数实时协同更新。

为了解决上述诸多挑战，设计了一套完整的面向异构混合算力的分布式并行训练方法，在后续章节进行详细介绍。

2　面向异构算力的分布式并行训练方法

2.1　实现流程

为实现同一训练任务在异构算力集群的任务拆解和协同执行，本文引入非均匀计算任务切分、最优并行策略预测、数据及任务协同等关键技术研究，综合分析不同算力集群的计算能力，进行训练任务与集群计算能力匹配，并实时完成训练中间值同步。

异构混合训练技术实现流程如图 1 所示，本文将在介绍总体技术架构基础上，分别阐述各关键技术设计方案。异构混合训练技术输入为特定训练任务，在分析训练模型结构以及异构算力集群基本情况后，采用异构混合训练非均匀计算任务切分算法进行计算能力评估、算力隔离及任务拆解，异构非均匀切分性能预测系统综合接收非均匀计算任务切分算法策略、单节点 AI 加速器硬件标称值、神经网络模型结构，模拟分析得出异构混合训练最优并行策略预测值，并通过混合训练框架将训练任务进行分布式并行分解执行。由于分布式训练过程中会产生一系列中间值（包括梯度、优化器状态、权重等），需要实时进行任务及数据传输协同，本文也定义了一套异构混合训练通信技术，实现异构场景下分布式训练中间值的实时同步更新，并在流程最后输出训练模型。

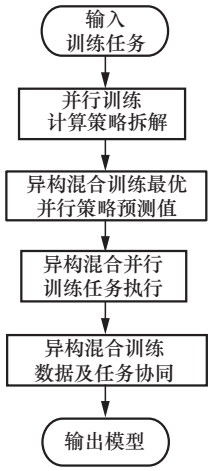


图 1　异构混合训练技术实现流程

2.2　技术架构

异构混合分布式训练技术总体架构如图 2 所示，由模型层、混合训练框架层、混合训练通信层、硬件层组成，通过框架层实现异构混合分布式并行训练计算策略拆解及最优性能预测策略执行，通过混合训练通信层实现异构 AI 加速器在训练任务执行过程中的数据传输协同。

异构混合分布式训练技术总体架构关键技术层说明如下。

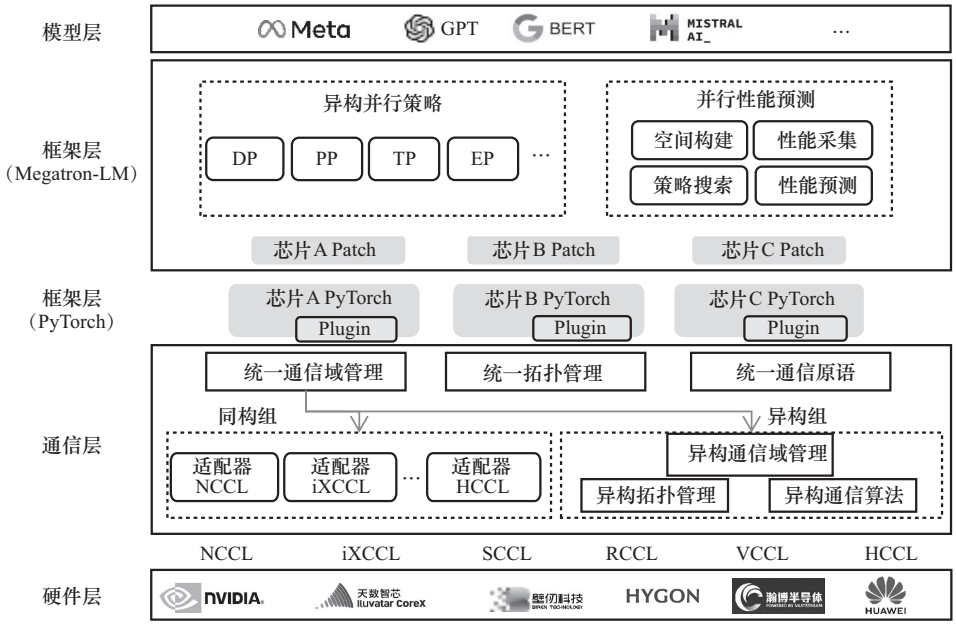


图2　异构混合分布式训练技术总体架构

2.2.1　混合训练框架层

异构混合训练框架层用于实现模型训练任务并行化拆解，在数据、模型等维度进行任务切分，切分后的任务与适配算力映射，实现训练过程的分布式并行处理。

由于异构混合训练需将切分任务映射到不同类型、规模的异构算力集群，当前业界所使用的分布式训练框架仅适用于同构算力集群环境，无法复用在异构混合训练场景中，因此本文方法在框架层设计了可适配异构算力集群的异构混训框架，基于业界主流Megatron分布式加速框架，结合异构混合训练非均匀计算任务切分算法进行非均匀并行能力改造，实现数据、模型等维度的非均匀任务切分。同时为实现非均匀切分性能最优，在框架层设计了异构非均匀切分性能预测工具，在给定模型规模及集群规模条件下，通过策略生成算法以及性能模拟工具分析生成适用该模型、集群的最优非均匀切分策略。

2.2.2　混合训练通信层

异构混合训练通信机制协同异构混合训练框架层，在框架层非均匀切分基础上进行并行训练进程的计算过程中间值传输同步，并基于异构混合训练任务调度映射策略实现调用智算硬件后端通信能力。

由于不同异构AI加速器在通信协议、互联方式、通信策略等层面存在着较大的区别，因此本文方法设计了一套数据传输通信层，通过统一通信组件层处理全局调度与通信分解，并由异构通信组件层处理跨节点异构通信，通过设备适配器层调用优化的同构通信库处理节点内或同构设备间的通信，解决各类AI加速器在通信机制层面的差异，实现异构混合训练通信能力。

2.3　关键技术

2.3.1　异构混合训练非均匀计算任务切分算法

混合分布式并行训练非均匀计算任务切分算法对齐同构集群训练方法，包括数据并行、张量并行、流水线并行等策略算法。本文对原始并行训练策略算法进行非均匀切分优化，下面以流水线并行、数据并行算法优化为例进行说明，其他并行切分策略可同理类推。

面对同一训练任务下的深度神经网络，由于混合分布式并行技术算力集群是异构的，需对隔离后不同的算力子集群放置不同的子训练任务（包括子神经网络层、子训练数据集等），基于不同子训练任务的计算量差异，对计算需求较大的子任务放置在算力较高的子集群中，将计算需求较小的子任务放置在算力较低的子集群中，最终实现混合分布式训练任务切分负载均衡。

1) 非均匀流水线并行技术

传统流水线并行训练过程可以分为预热阶段、稳态阶段[8]。预热阶段由于不同流水线层处理数据微批次存在顺序，因此存在气泡时间且整体呈现“倒三角”形态；稳态阶段流水线并行执行效率最快，

AI加速器计算呈现趋零等待现象。传统流水线并行计算执行过程如图3所示。由图3可以看出，气泡时间存在于预热阶段，稳态阶段AI加速器计算等待时间趋零，此时均匀切分可以达到较优并行性能。

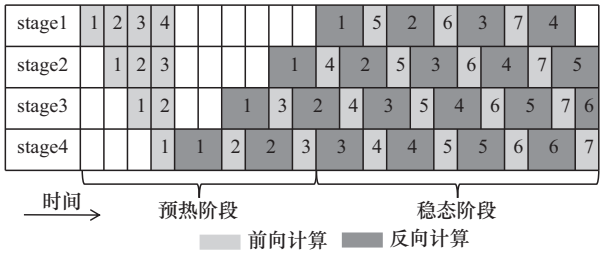


图3　传统流水线并行计算执行过程

当考虑资源池AI加速器算力不同时，由于芯片计算速度存在差异，传统均匀切分方法不再适配异构场景。假设在特定训练任务下，资源池共计有4张AI加速器，每张AI加速器分配1个流水线stage，其中AI加速器1、2算力分别是3、4算力的2倍，进行分布式训练执行效果分析，整体预热阶段计算时间较长，计算能力高的芯片需要等待计算能力低的芯片，气泡时间呈现较长“倒三角”形态。非均匀流水线并行计算执行过程如图4所示，由图4可以看出，预热阶段及稳态阶段均出现气泡时间，其中稳态阶段气泡时间需使用非均匀计算任务切分算法进行负载均衡；当AI加速器1、3进行第2个、第3个微批次数据的反向计算，AI加速器4进行第4个微批次数据正向计算时，流水线并行训练过程进入稳态阶段。

稳态阶段流水线并行执行效率最快，仅有部分或全部AI加速器计算呈现趋零等待现象，从图4可以看出，当进入稳态阶段，特定时刻仅有1个设备未参与计算，AI加速器3、4前、反向计算过程零等待。

由于2种设备计算能力存在差异，处于稳态阶段计算能力较强的芯片在训练过程中仍存在少量气泡时间，其气泡时间计算式为

$$T_{\text{bubble}} = T_d(K-1)(N+1) \tag{1}$$

其中，K代表芯片A与芯片B的算力比值，N代表反向计算时间与前向计算时间的比值，T_d为单微数据批次单层计算时间。例如，当芯片A与B算力相当时，稳态阶段不存在气泡时间，当反向计算时间为前向计算时间2倍，且算力比值为2时，气泡时间为$3T_d$。

本文方法在Megatron流水线并行交错式调度技术（1F1B，one forward one backward）基础上提出非均匀流水线并行技术。首先，在并行配置上根据不同AI加速器计算能力增加非均匀配置策略，即允许将不同PP stage配置在不同的AI加速器集群上，并根据异构集群各自的计算能力差异化地分配不同的网络层；其次，进行流水线气泡时间分析，将非均匀流水线并行技术的稳态阶段进行执行效率优化，压缩稳态阶段气泡时间，实现流水线非均匀切分负载均衡。

为了实现稳态阶段气泡时间压缩，可通过衡量大语言模型Transformer层的计算量[9]，结合芯片集群算力情况调节流水线并行配置，通过不同芯片在单网络层上的计算速度趋近统一，实现并行计算过程最优化问题。即

$$\begin{aligned} &\min\left(\left|\frac{T_{\text{A}}}{L_{\text{A}}}-\frac{T_{\text{B}}}{L_{\text{B}}}\right|\right)\\ &\text{s.t.}\quad P_{\text{A}}+P_{\text{B}}=P\\ &\qquad L_{\text{A}}P_{\text{A}}+L_{\text{B}}P_{\text{B}}=L \end{aligned} \tag{2}$$

其中，T_{A}和T_{B}代表芯片A和芯片B的算力，对于特定芯片，T_{A}和T_{B}是一个常数；L_{A}和L_{B}代表芯片A和芯片B被分配的层数；P_{A}和P_{B}代表芯片A和芯片B的流水线并行度；P代表流水线并行的维度；L代表模型总层数，对于特定模型，L是一个常数，例如LLaMA2-7B对应的层数为32，LLaMA2-70B对应的层数为80。

流水线并行非均匀切分策略说明如图5所示，

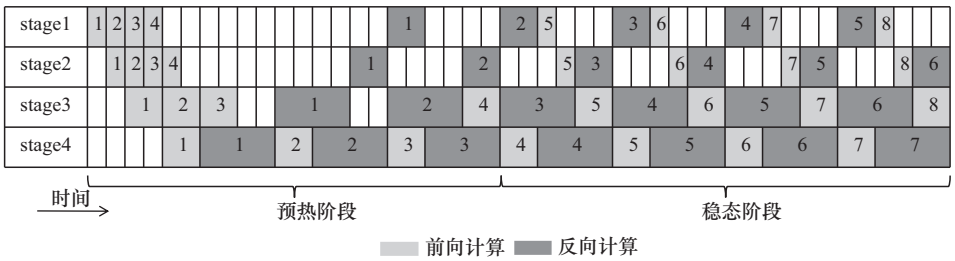


图4　非均匀流水线并行计算执行过程

其中，实线箭头为前向计算过程，虚线箭头为反向计算过程。将数据批次均匀切分为同样大小的micro batch，分别进行流水线并行计算，设定此时流水线并行度p=1，根据不同AI加速器的算力进行网络各层分配，其中Layer0和Layer1分配给芯片A处理，Layer2~Layer4分配给芯片B处理，Layer n分配给芯片N处理，完成流水线非均匀切分负载均衡。

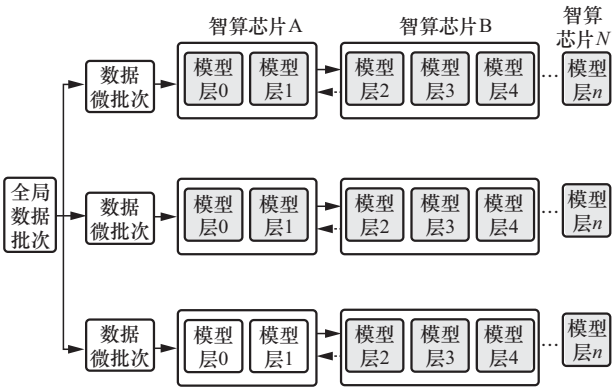


图5 流水线并行非均匀切分策略说明

2) 非均匀数据并行技术

在传统数据并行技术中，分布式并行策略需根据模型参数规模及集群AI加速器数量，调整切分到各AI加速器上的数据微批次大小，其中数据微批次大小采用均匀切分方式配置，并在反向计算阶段进行参数同步。

在异构混池训练场景，情况变得复杂。数据并行组内各芯片算力存在差异，若仍采用均匀切分方式配置数据微批次大小，则不同批次数据参与的计算无法同步完成，会出现较明显的通信等待现象，计算速度较快的设备需等待计算速度较慢的设备完成计算过程后，才会进行参数同步通信。

为平衡数据并行计算差异问题，本文首先提出数据微批次粗颗粒度非均匀配置方法，即需要根据算力情况分配等比例的数据微批次大小，计算式为

$$\min\left(\left|\frac{T_{\mathrm{A}}}{T_{\mathrm{B}}}-\frac{M_{\mathrm{A}}}{M_{\mathrm{B}}}\right|\right) \tag{3}$$

其中，M_{A}和M_{B}代表芯片A和芯片B分别配置的micro batch size。当min值越接近0，通信等待时间就越少；当min=0时，数据微批次大小与各芯片算力实现等比例配置，理论上解决了数据并行计算不平衡问题。

混合并行训练执行时间分析如图6所示，其中图6(a)和图6(b)分别为芯片A和芯片B的并行训练分析，假设芯片A计算能力为B的$\frac{1}{2}$，根据第一步粗颗粒度配置方法，A分配的数据微批次大小为B的$\frac{1}{2}$。从图6可以看出，理论上两批数据的有效计算时间与气泡时间可实现同步。

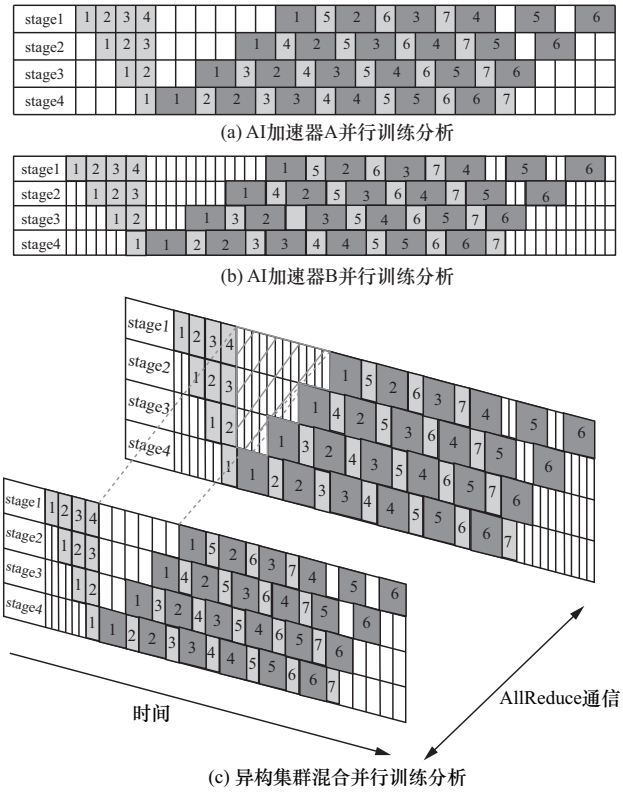


图6 混合并行训练执行时间分析

然而，实际工程场景下min值很难实现趋零配置，即有效计算时间与气泡时间难以通过粗颗粒度方法实现完全同步。为了进一步解决实际工程场景异构混池条件下数据并行计算不平衡问题，需结合流水线并行气泡时间分析，在上述方案基础上做细粒度调优设计，即压缩流水线并行计算过程预热阶段气泡时间，使其进行数据并行AllReduce通信的时间保持同步。

本文在微批次大小非均匀调节基础上，引入微批次数量调节机制，将数据微批次较大的计算过程拆解为多个单位数据微批次多次送入流水线，通过将多个微批次数据间的P2P通信时间与计算时间重叠，压缩流水线气泡时间，使不同芯片在单数据微批次上的计算速度趋近统一，实现不同芯片计算负载均衡。计算式为

$$\min\left(\left|\frac{T_A}{M_A}-\frac{T_B}{M_B}\right|\right)$$
$$\text{s.t.}\quad d_A+d_B=d$$
$$\mathrm{MS}_A m_A d_A+\mathrm{MS}_B m_B d_B=G \tag{4}$$

其中，MS_A和MS_B分别代表芯片A、芯片B在单位数据微批次下的micro batch size，m为数据批次数量，G为global batch size；d_A和d_B代表芯片A、芯片B数据并行度，即分别有多少片芯片组成一个数据并行组。

数据并行非均匀切分策略说明如图7所示，其中，实线箭头为前向计算过程，虚线箭头为反向计算过程，A, B, … ,N分别为不同厂商AI加速器，假设$d=1$，$m=N$，此时每种类型异构芯片处理一个数据切片micro batch，micro batch size根据不同芯片算力进行匹配设置，分别为M_A,M_B, … ,M_N。

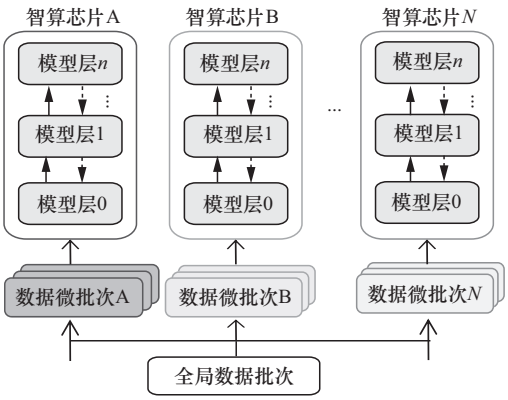


图7　数据并行非均匀切分策略说明

2.3.2　异构非均匀切分性能预测技术

近年来，训练集群规模持续增长，基于人工经验手动调优并实测的传统方案难以满足大规模异构分布式训练的需求。当集群从同构GPU组成变为异构GPU组成时，并行策略的搜索是一个NP hard问题[10]，搜索空间的扩增难以在有限的时间内搜索得到高效的并行策略。为解决上述问题，本文提出异构非均匀切分性能预测技术。该技术分为3个部分。

并行策略搜索空间构建：在数据并行、张量并行、流水线并行等多维度策略组合和重计算等优化参数基础上构建多维参数空间。

性能及仿真预测：以基础性能采集结果作为先验信息，结合理论计算，完成性能及访存预测功能。

搜索及剪枝策略：设计两阶段搜索机制。首先将参数空间建模为树结构，在参数空间树上以深度优先搜索的方式生成候选配置；然后在精调阶段设计动态规划算法，寻找流水线并行非均匀切分等最优配置策略。

异构非均匀切分性能预测技术架构如图8所示。性能预测系统包括基础性能采集、性能及访存预测、搜索策略及剪枝、参数空间构建等组成部分。异构非均匀切分性能预测系统接收神经网络模型结构、单节点AI加速器硬件标称值、集群实际通信带宽和训练框架适配的切分算法。预测系统尝试搜索多种并行策略，对所有可行的策略进行性能预测并比较，输出最优训练配置文件到异构混合训练框架中。

1) 并行策略搜索空间构建

在数据并行、张量并行、流水线并行等多维度策略组合和重计算等优化参数基础上构建多维参数空间。考虑的配置参数总结如下。

tp：张量并行参数，通常有1、2、4、8等预选配置。

dp：数据并行参数。

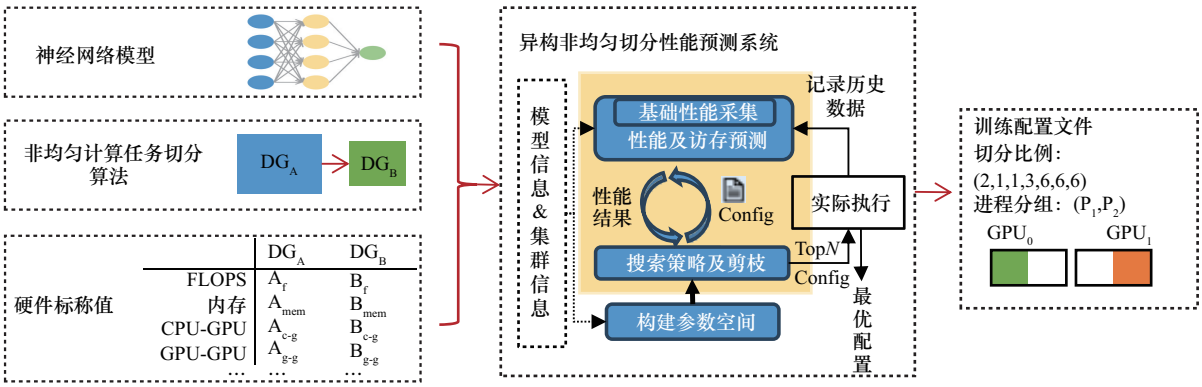


图8　异构非均匀切分性能预测技术架构

pp：流水线并行参数。

num_layers_of_each_pp：每个stage分配的层数。

rc_of_each_pp：重计算参数。

2) 性能及仿真预测

本文方法需要为不同参数组合提供性能及访存预测功能，通过在基础数据采集基础上，利用性能及访存预测模型模拟端到端整体性能。

首先，本文方案通过Profiling获取目标网络的单网络层在不同TP参数配置、不同硬件后端的前/反向执行耗时等基础数据。这些基础数据后续将会用于端到端训练时间的计算。值得注意的是，这种基于网络层粗颗粒度采集的数据包含2个部分：计算时延和TP策略引起的通信开销。

结合2.3.1节的讨论，异构流水线迭代时间分析如图9所示。假设每个stage均分配相同层数L，此时每个格子代表了L层网络组成stage的耗时，即L*单层网络采集时延。

结合图9分析可得出网络端到端计算时延和TP下的通信时延计算方法，即性能及访存预测模型，如式(5)所示，其中d代表不同的芯片类型。

$$T_{(\text{comp}+\text{tp\_comm})}[\text{tp}] = T_{\text{warmup}}[\text{tp},d] + T_{\text{steady}}[\text{tp},d] + T_{\text{cooldown}}[\text{tp},d] \tag{5}$$

对于DP和PP策略，主要考虑AllReduce和P2P通信时延。统计出计算流程中出现的参数传输需求，实测这些传输需求在多个通信原语下的耗时。综合以上讨论，端到端总耗时计算式为

$$T_{\text{cost}} = T_{(\text{comp}+\text{tp\_comm})}[\text{tp}] + T_{\text{pp}}[\text{tp}] + T_{\text{all\_reduce}}[\text{dp}] \tag{6}$$

3) 搜索及剪枝策略

本文基于前文所述几种并行策略和优化参数的不同特点设计两层次搜索机制。

第一层次搜索通过异构并行策略寻优算法，获取DP、TP和PP这3个参数值，具体实现如算法1所示。在集群资源量、TP备选值的约束下，这3个参数的组合量可控，本文方法首先通过深度优先搜索（DFS,depth first search）的方式，遍历由pp→dp→tp顺序组成的策略树[11]，获取[pp, dp, tp]参数组合。

算法1 异构并行策略寻优算法

输入 模型结构model，集群配置cluster

输出 最优的并行策略

1)parallelStrategys ← autoParallel (model, cluster) // 构建并行策略空间

2)Initialize minIterTime ← Inf // 初始化最小迭代时延

3)for each parallelStrategy_tp, dp, pp in DFS(parallelStrategys) // 在并行参数空间中按照DFS的策略遍历

4)parallelStrategy, time = FineTuneStrategy([pp, dp, tp]) // 基于非均匀层分配算法做策略微调

5) if time < minIterTime && DelectAndAlleviateOOM([pp, dp, tp]) == SUCCESS // 性能预测及判断

6) minIterTime ← time // 更新最小迭代时延

7) bestParallelStrategy ← parallelStrategy //更新最小时延下的最佳并行策略

8) end if

9) end for

10) return minIterTime, bestParallelStrategy // 返回最小最佳并行策略及最小时延

在并行策略的搜索过程中，由于搜索空间较大，为了在有限时间内得到有效的参数组合方案，

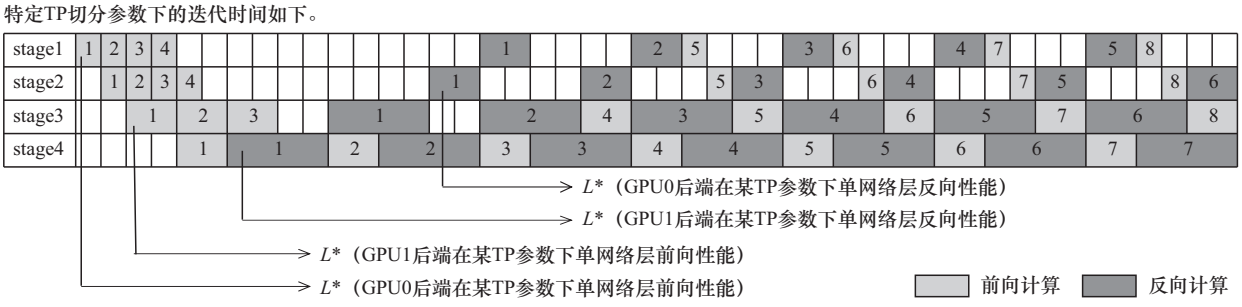


图9 异构流水线迭代时间分析

有必要对庞大的搜索空间进一步剪枝缩减搜索空间。本文采用贪心剪枝寻优策略[12]，舍弃内存溢出（OOM，out of memory）等非法配置。如算法 1 中的 DelectAndAlleviateOOM 模块所示，在参数寻优过程中，首先会对当前配置下的 PP stage 做显存占用量预测，然后基于贪心寻优的策略做微调；如果出现 OOM 的情况，会逐步增加重计算层，直到缓解 OOM。如果所有的层都做了重计算，依然无法缓解 OOM，那么这组配置将被舍弃。

为了解决 PP 并行策略下的负载不平衡瓶颈，本文在第二层次搜索中提出了非均匀层分配算法，如算法 2 所示。该算法微调每个流水线阶段分配的具体网络层数，可进一步获取当前[pp, dp, tp]组合下的最佳微调参数。

算法 2　非均匀层分配算法

输入　tp,dp,pp，模型层数 N，每层时延 layer_time，微批次 B

初始化：

T_Edge[pp][N] ← 0 // edge time，即非稳态时间(warmup ,cooldown)

T_Middle[pp][N] ← 0 // Middle phase duration，即稳态时间（steady）

T_all_reduce[pp][N] ← 0 // all_reduce communication time

输出　微调后最优策略(parallelStrategy)

1)for each stage i from 0 to (pp –1): // 获取当前 stage 下

2)　for start_op in reverse (N–1 down to 1): // 以 start_op 为起点时的最优值

3)　　T_min ← ∞

4)　　optimal_end_op ← –1// 遍历当前 start_op 对应的 end_op

5)　　for end_op in reverse(N–1 down to start_op): // 获取当前流水线阶段的时延

6)　　t_stage = (end_op − start_op + 1)× profiled_layer_time // 获取加入当前流水线阶段后的非稳态时间

7)　　T_Edge_tmp = T_Edge[i–1]×[end_op+1] + t_stage // 获取加入当前流水线阶段后的稳态时间，假设本阶段没有被其他流水线阻塞住

8)　　T_Middle_tmp = T_Edge_tmp + (B – PP + i – 2) × t_stage

9)　　if (t_stage < T_Middle[i − 1] [end_op+1]) // 若本阶段被其他流水线阻塞住，需插入气泡等待，更正稳态时间。

10)　　T_Middle_tmp = T_Middle[i − 1] [end_op+1] // DP 组通信时间预估

11)　　T_all_reduce_tmp = max(t_all_reduce, T_all_reduce[i–1] [end_op+1]) // 获取总时延

12)　　total_time = T_Edge_tmp + T_Middle_tmp + T_all_reduce_tmp

13)　　if total_time < T_min: // 更新最小时延和对应并行策略参数

14)　　T_min ← total_time

15)　　optimal_end_op ← end_op

16)　　T_Edge[i][start_op]←T_Edge_tmp

17)　　T_Middle[i] [start_op] ←T_Middle_tmp

18)　　T_all_reduce[i][start_op]←T_ all_reduce_tmp

对于某一流水线阶段，分配给它的网络层是连续的，该特性使网络层分配问题的解空间具有最优性和多项式运行时间[13]，动态规划（DP,dynamic programming）是解决该类问题的一种有效算法。因此在算法 2 中本文提出动态规划方法处理 stage 间网络层分配问题。

在算法 2 中，start_op 和 end_op 指某个 stage 的起始网络层和结束网络层。算法遍历 start_op 和 end_op 参数获取最优解。值得注意的是，稳态下[start_op, end_op]的实际耗时需做一次仿真结果分析，如果[start_op, end_op]仿真的理论迭代时间小于历史记录的最小耗时 stage，意味着该 stage 被其他流水线阶段阻塞住，必须插入气泡进行等待，保持迭代时间与最小耗时 stage 一致。

本文搜索算法复杂度主要来自 2 个方面：首先是集群规模，即 AI 加速器的数量 M，其直接影响[pp, dp, tp]的组合数；其次是网络规模，即非均匀层分配算法中针对模型层数 N 的两层循环。综合分析，整体算法复杂度为 $O\left(MN^2\right)$。

最后输出最优训练配置文件，其中包含最优切分比例和进程分组信息。在图 8 示例中，根据性能

预测系统分析结果，非均匀流水线并行在GPU0、GPU1上切分的size分别为2、3，对应GPU0流水线并行阶段1、2分别放置1层Layer，GPU1流水线并行阶段1、2、3分别放置6层Layer。

2.3.3　异构混合训练通信技术

当前主流硬件厂商（如NVIDIA、Intel、AMD等）均提供针对其硬件优化的专用集合通信库（如NCCL、Gloo、Radeon集合通信库（RCCL，radeon collective communication library）等）。然而，在混合集群场景，异构硬件间的高效集合通信面临极大挑战，不同厂商通信库在底层流程设计、通信算法实现等方面存在差异，难以实现跨厂商硬件的通信协同。

1) 统一异构集合通信库架构设计

为解决上述挑战，本文提出一种统一异构通信库架构，如图10所示。该架构采用分层设计思想，包括以下层级。

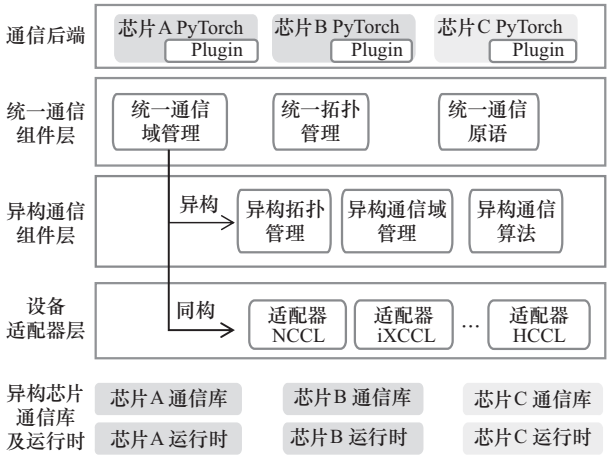


图10　统一异构集合通信库架构

①通信后端层：负责对接上层框架（如PyTorch），提供统一的通信后端调用接口。以对接PyTorch为例，该层实现自定义的ProcessGroup，使用PyTorch的collective communication API钩子机制，覆盖默认的通信操作。

②统一通信组件层：作为核心调度层，通过接收通信后端统一通信与管理、统一拓扑管理和统一通信原语指令，实现对设备适配器和异构通信组件的调度。

③异构通信组件层：处理异构节点间通信。支持不同集群芯片通过Gloo或者Proxy Service/Proxy Progress调用RDMA的方式实现GPU远程直接内存访问（GDR, GPU direct RDMA）互通。

④设备适配器层：封装并适配各AI加速器同构通信库（如NCCL、RCCL）及其运行时。适配层提供了通信任务从统一异构通信框架到各厂商通信库的透传。实现细节是对各厂商集合通信库开放的3类接口做一层封装，包括：通信组管理接口，如通信域的初始化和构建；集合通信原语接口，如AllReduce、AllGather；运行时接口，如不同AI加速器device显存的申请接口。

该架构的核心机制在于通过统一管理、分层解耦与协同调度，实现异构集合通信。具体而言，将异构通信问题拆解为：由统一通信组件层处理全局调度与通信分解；由异构通信组件层处理跨节点异构通信；由设备适配器层调用优化的同构通信库处理节点内或同构设备间的通信。

本文方法选择在现有同构通信库之上进行封装，而非强制统一底层通信库实现标准，主要基于以下考量：各厂商的通信库深度集成了针对其特定硬件架构（如NVLink、Infinity Fabric）和底层链路的优化策略。若试图统一底层实现标准，不仅改造工作量巨大，更可能破坏或削弱这些经过深度优化的性能优势。本文封装架构有效解耦了异构通信适配与底层通信优化，使上层应用不需要感知底层硬件差异，并且能够最大化复用并发挥各芯片通信库的固有性能优势，从而在保证异构兼容性的同时，维持接近原生同构通信的性能水平。

2) 统一通信原语设计与协同调度机制

为具体阐述同构异构拆解协同调度机制，本节以统一通信原语设计为例进行说明。现有同构通信库在处理多机通信时，常采用“机内–机间–机内”的分层通信策略[14]。借鉴此思想，本文设计的统一通信原语将异构集合通信拆解为“同构–异构–同构”的3个阶段模型。

①同构通信（节点内/同构设备间）：由各节点内针对特定硬件优化的同构通信库（如NCCL、RCCL）执行。

②异构通信（跨节点异构设备间）：由异构通信组件负责实现跨智算集群节点的通信。

③同构通信（节点内/同构设备间）：再次由同构节点的同构通信库完成最终数据处理。

与同构集群通常具备高速、低时延、规则拓扑

的内部网络（如 InfiniBand、高速以太网）相比，异构机间互联的网络拓扑往往更复杂、跳数更高，带宽或时延性能可能受限。鉴于一次完整的集合通信总耗时是上述 3 个阶段通信时间的累加，为最小化异构层通信时延对整体性能的影响，异构通信组件采用基础的 Send/Recv 点对点通信原语作为核心传输机制，实现异构层通信策略选择。这种设计方法基于以下考量。

①简化与通用性：Send/Recv 是最基础、兼容性最广的通信原语，易于在不同硬件平台和网络协议上实现高效、可靠的适配。

②适配非均匀切分策略：结合异构混合训练非均匀计算任务切分算法工程实践情况，异构流水线并行策略目前是工程上最易用的异构非均匀切分策略，异构 Send/Recv 能够满足其数据传输需求。

除了基础点对点通信，为适配异构数据并行策略能力，本文方法针对 AllReduce 等集合通信操作进行了 3 个阶段模型能力设计，集合通信算子语义拆分如图 11 所示，展示了异构 DP 非均匀切分所需的 AllReduce 操作 3 个阶段拆解流程。

①Reduce（同构）：在每个同构域内，由本地同构通信库执行 Reduce 操作，将所有 rank 上的数据聚合到指定的主 rank（如 rank0）上。

②Send/Recv（异构）：异构通信组件负责以异构主 rank 为 send 设备，另一通信域的非主 rank 为 recv 设备，传输聚合的数据。

③AllReduce（同构）：调用各同构域 AllReduce,实现全局 AllReduce。

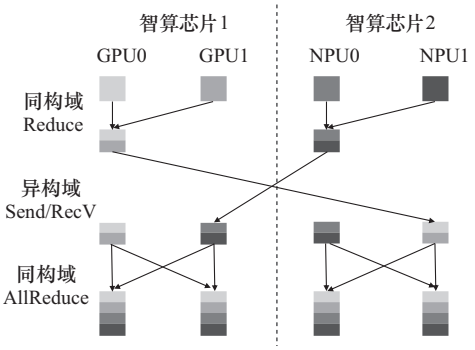


图 11　集合通信算子语义拆分

3　技术验证

3.1　实验条件

为分析验证异构混合训练可行性及训练效率，本文实验在固定模型训练步数条件下，分别测试单一类型同构算力集群、混合算力集群条件下的并行训练效果。

实验选用 Meta 开源的 LLaMA2 7B、LLaMA2 13B 模型，数据集为 Redpajama(306B Token)，采用 Nvidia H100、天数智芯 BI-V150、壁仞壁砺 106B 作为实验所用算力资源池，训练框架分别使用 Megatron LM、异构混训框架。

在单一类型同构算力集群实验中，分别面向 Nvidia H100、天数智芯 BI-V150、壁仞壁砺 106B 同构集群分别测试 LLaMA2 7B、13B 模型训练效果，训练框架采用 Megatron LM。单一类型同构算力集群如图 12 所示，其中，在 Nvidia H100 集群中，共计 3 个节点 24 张卡；BI-V150 集群中，共计 3 个节点 48 张卡；壁砺 106B 集群中，共计 1 个节点 8 张卡。

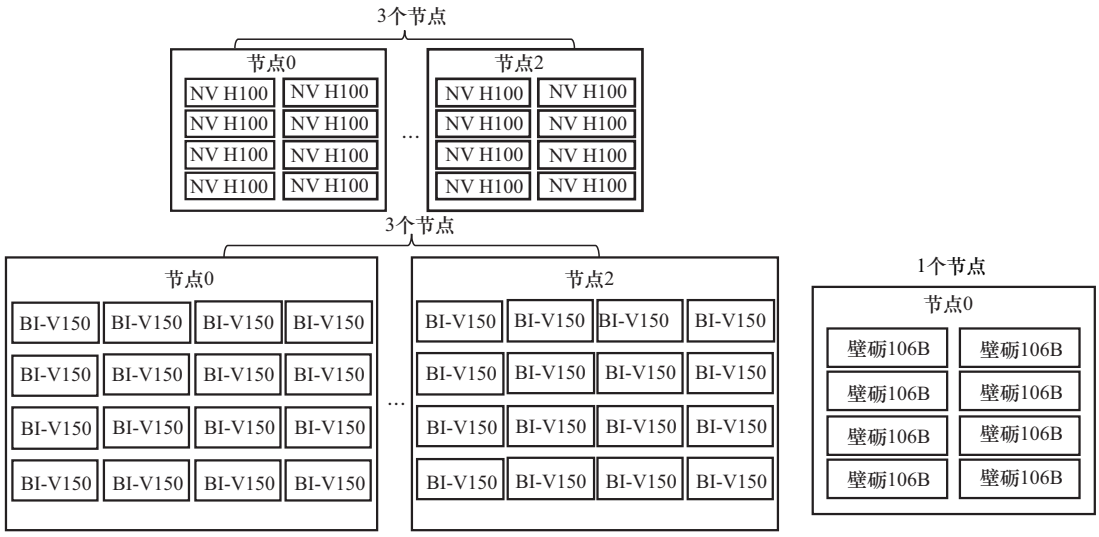


图 12　单一类型同构算力集群

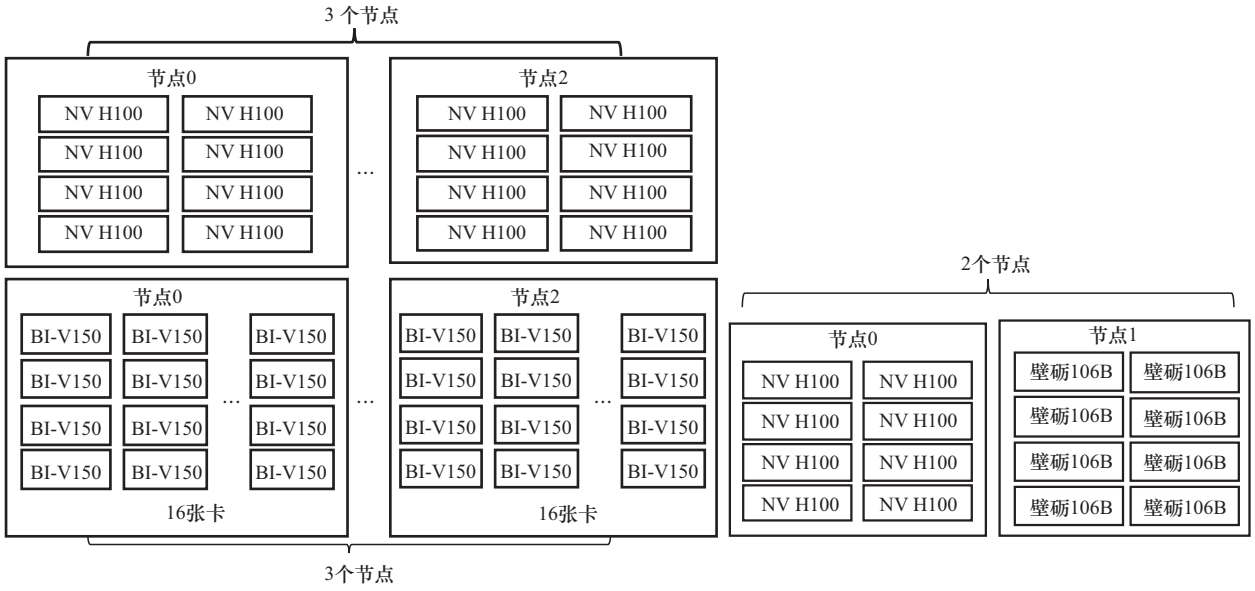


图 13 异构混合算力集群

在异构混合算力集群实验中，面向 Nvidia H100、天数智芯 BI-V150、壁仞壁砺 106B 组成的混合算力资源池测试 LLaMA2 7B、13B 模型训练效果，训练框架采用本文异构混训框架。异构混合算力集群如图 13 所示，其中，在 Nvidia H100+BI-V150 集群中，NV、天数集群各 3 个节点，共计 6 个节点 72 张卡；在 Nvidia H100+壁砺 106B 集群中，NV、壁仞集群各 1 个节点，共计 2 介节点 16 张卡。3 类 AI 加速器集群间采用 IB 交换机互联，网络带宽分别为 400 Gbit/s、200 Gbit/s、200 Gbit/s。

3.2 实验分析

3.2.1 吞吐率

本文实验采用集群吞吐率 TPS（即系统每秒生成的 token 数量）衡量计算能力，集群类型包括同类型芯片组成的同构算力集群、不同类型芯片组成的异构算力集群，计算式为

$$T_x = 1\,000\frac{SG}{P} \tag{7}$$

其中，S 为序列长度，G 为全局数据批次大小，P 为特定类型芯片集群训练单步执行时间（单位为 ms）。若在训练迭代步数相同条件下，芯片 A 集群吞吐率比芯片 B 集群吞吐率高，说明芯片 A 集群具有更好的计算能力。

LLaMA2 7B、13B 训练吞吐率如图 14 所示。由图 14 可以看出，在 LLaMA2 7B、LLaMA2 13B 模型训练过程中，单一类型 Nvidia、BI-V150、壁砺 106B 吞吐率各异，当进行混合资源训练后无论是 NV GPU 与 BI-V150 还是 NV GPU 与壁砺 106B，混合集群训练吞吐率均优于单类型集群训练结果，说明采用本文方法进行异构混合训练能够提高集群训练计算效率。

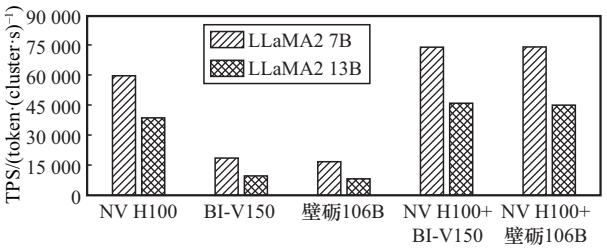


图 14 LLaMA2 7B、13B 训练吞吐率

3.2.2 算量比

本文实验采用算量比 N 作为衡量非均匀切分负载均衡指标，计算式为

$$N = \frac{Q_{\mathrm{A}}}{Q_{\mathrm{B}}}$$
$$\text{s.t.}\quad Q_i = \frac{T_i}{K_i} \tag{8}$$

其中，i 为集群类别，T_i 为芯片 i 的算力，K_i 为芯片 i 被分配的神经网络层数（流水线并行）或微数据批次（数据并行），Q_i 表示 i 类芯片单位计算维度算力情况，N 为 2 种类型单位计算维度算力比值。当 $N > 1$ 时，说明集群 A 计算量大于集群 B；当 $N < 1$ 时，集群 A 计算量小于集群 B。若要实现负载均衡，N 应趋近于 1，说明此时非均匀切分并行策略调试比例可充分发挥 2 种异构集群计算能力。

为引入多指标综合衡量负载均衡效果，同时观测训练过程中显存占用变化情况进行综合分析。针对非均匀流水线并行、数据并行技术，为不同类型AI加速器配置不同神经网络层及微数据批次，特定芯片在计算量较大的任务下呈现较大显存占用，在计算量较小的任务下呈现较小显存占用，则说明该芯片面向特定非均匀并行技术可实现有效配置（计算任务负载均衡）。

本文实验在保证训练迭代步数相同的情况下进行算量比调整，LLaMA2 7B模型NV H100/BI-V150算量比效果如图15所示，LLaMA2 7B模型NV H100/壁砺106B算量比效果如图16所示。在NV H100+BI-V150混合算力环境下，当算量比为1时，加速比可达94.5%；在NV H100+壁砺106B混合算力环境下，当算量比为1.5时，加速比可达96.9%，证明当训练算量比趋近1时，异构集群计算能力最好，可实现负载均衡。

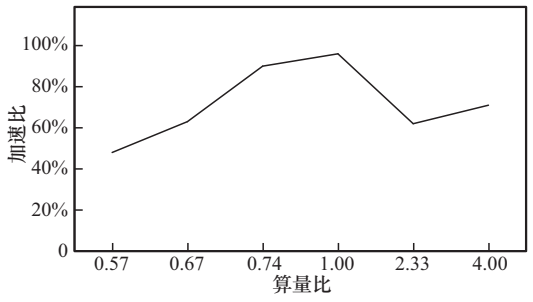


图15　LLaMA2 7B模型NV H100+BI-V150算量比效果

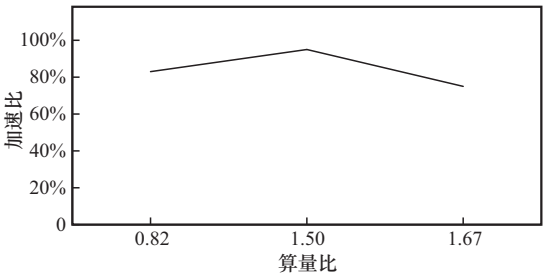


图16　LLaMA2 7B模型NV H100+壁砺106B算量比效果

同时，根据显存占用情况观测，在特定非均匀流水线并行配置条件下，当NV H100配置a的PP层数小于配置b，BI-V150、壁砺106B配置a的PP层数大于配置b，LLaMA2 7B非均匀流水线并行训练显存占用变化如图17所示，NV H100吞吐率在配置a条件下小于配置b，BI-V150、壁砺106B吞吐率在配置a条件下大于配置b。

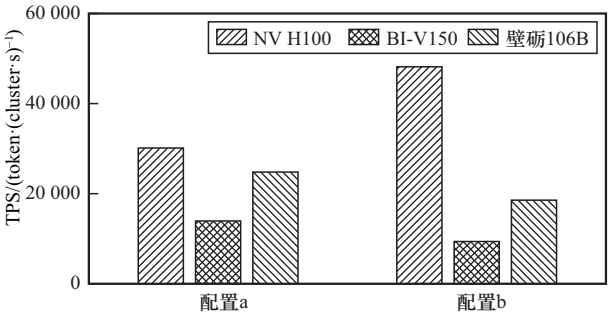


图17　LLaMA2 7B非均匀流水线并行训练显存占用变化

在特定非均匀数据并行配置条件下，当NV H100配置a的数据微批次小于配置b，BI-V150、壁砺106B配置a的数据微批次大于配置b。LLaMA2 7B非均匀数据并行训练显存占用变化如图18所示，NV H100吞吐率在配置a条件下小于配置b，BI-V150、壁砺106B吞吐率在配置a条件下大于配置b。

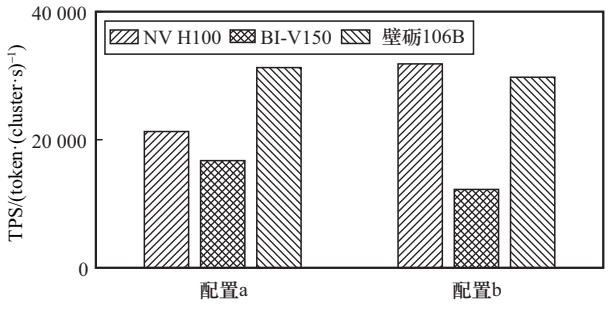


图18　LLaMA2 7B非均匀数据并行训练显存占用变化

3.2.3　通信功能及效率

异构统一通信效率主要考虑两类通信原语：点对点通信（Send/Recv）和复杂集合通信原语。本文实验选取具有代表性的Send/Recv和AllReduce进行通信功能测试和通信带宽测试。Send/Recv算子功能及通信带宽如图19所示，AllReduce算子功能及通信带宽如图20所示。

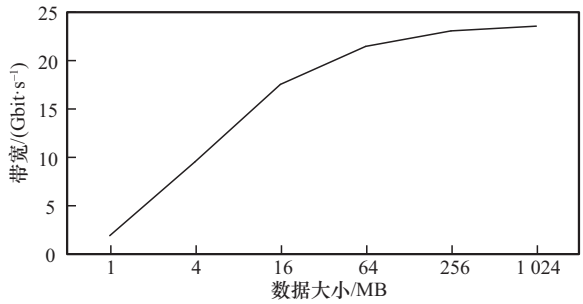


图19　Send/Recv算子功能及通信带宽

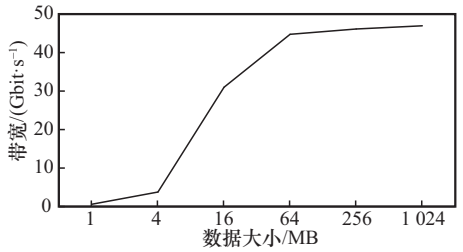


图20 AllReduce算子功能及通信带宽

通过实验可知，本文异构统一通信集合通信库可以达成多种异构AI加速器传输训练参数的目标。同时，随着通信数据量的提升，带宽可以逐步提升直至稳定状态。

3.2.4 性能预测搜索效率及准确度

1) 搜索效率

实验选取LLaMA2 7B模型对搜索效率进行验证。首先，约束AI加速器数量为48卡，测试随模型网络层数的增长搜索时间变化情况，随着网络层数增长搜索时间变化情况如图21所示。实验表明，随着网络层数的增长，搜索时间控制在线性增长的范围，因为网络层分配过程中不合理的策略会被剪枝。

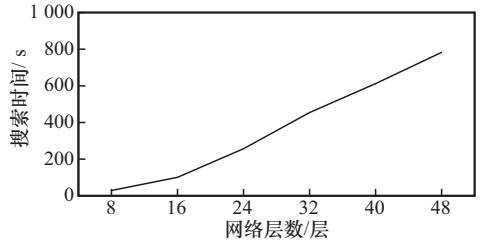


图21 随着网络层数增长搜索时间变化情况

其次，约束网络层数为32层，随着AI加速器增长搜索时间变化情况如图22所示。由图22可以看出，随着集群规模的增大，搜索时间依然控制在缓慢线性增长的范围。另外，某些集群的并行参数组合不具备亲和性，可搜索到的并行配置组合较少，搜索时间反而会下降。

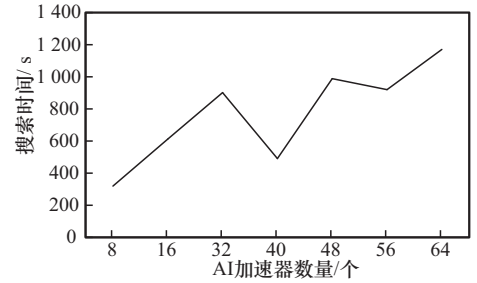


图22 随着AI加速器增长搜索时间变化情况

综上，在集群规模和网络规模扩大的过程中，本文方法的搜索时间可以保证在多项式时间之内解决，搜索效率稳定可控。

2) 搜索准确度

搜索准确度指本文方法输出的推荐并行参数训练端到端耗时与手动调参实际能找到的最佳并行参数训练端到端性能之间的差异。计算式为

$$E_{\text{acc}} = 1 - \left| \frac{T_{\text{auto}} - T_{\text{base}}}{T_{\text{base}}} \right| \tag{9}$$

其中，T_{auto}为本文方法推荐的并行参数运行耗时，T_{base}为手动调参找到的实际最优并行参数运行耗时。本文选取模型LLaMA2 7B，调整网络层数为24层、28层、32层、34层、36层，测试随网络层数变化搜索准确度的数据，LLaMA2 7B推荐并行配置实际端到端耗时与手动调优对比情况如图23所示。

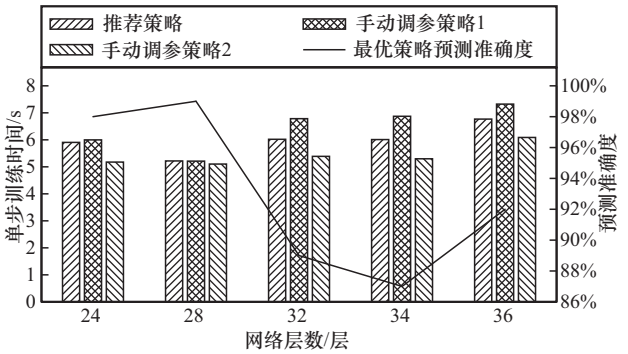


图23 LLaMA2 7B推荐并行配置实际端到端耗时与手动调优对比情况

根据实验结果，当模型层数较小时，搜索预测准确度较高，基本可达98%以上，伴随模型层数的增加，搜索准确度有一定波动，但基本不低于87%。综合各类模型层数，性能预测搜索模块准确度平均可达93%，具有较好的并行策略预测能力。

3.2.5 加速比

加速比R_{i}作为衡量并行计算性能损耗的指标，其计算式为

$$R_{\text{i}} = \frac{T_h}{\sum_{1}^{\alpha} T_i},\ R \in [0,1] \tag{10}$$

其中，R_i为混合集群X加速比；T_h为混训实测集群吞吐TPS；$\sum_{1}^{\alpha} T_i$为混训合池性能理论吞吐TPS，其中T_i为特定AI加速器在单一集群上的吞吐值，混

合集群包含α类AI加速器。

当实测合池集群吞吐T_h接近理论吞吐$\sum_{1}^{\alpha}T_i$时，说明采用异构混合训练后性能相比各集群单一性能总和来说，计算效率损耗较小，即异构混合训练并行策略优化效果越好。

LLaMA2 7B、13B 训练效果比较分析如图 24 所示。由图 24 可以看出，在 LLaMA2 7B、LLaMA2 13B 模型训练场景下 NV H100+BI-V150、NV H100+壁砺 106B 两类异构混合集群加速比均超 90%，计算损耗在 4%~5%，表明异构混合训练性能相比各个集群上单测性能总和来说，效率损耗小，且并行策略优化效果也较佳。

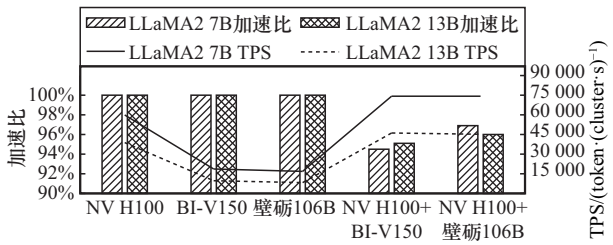


图24　LLaMA2 7B、13B 训练效果比较分析

3.2.6　收敛精度

收敛精度是衡量对比不同系统训练精度的主要指标，其通过计算不同系统在特定训练迭代的损失值误差，评估训练收敛准确性。计算式为

$$E_{\text{abs}} = \left| x_i - y_i \right| \tag{11}$$

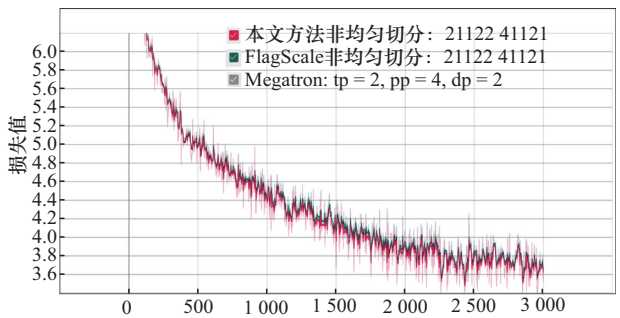
$$E_{\text{rel}} = \left| \frac{x_i - y_i}{y_i} \right| \tag{12}$$

其中，E_{abs}为绝对误差，E_{rel}为相对误差，x_i表示实测值，y_i表示基准值。绝对误差反映不同系统损失值偏差的实际大小，相对误差反映不同系统损失值偏差比例。

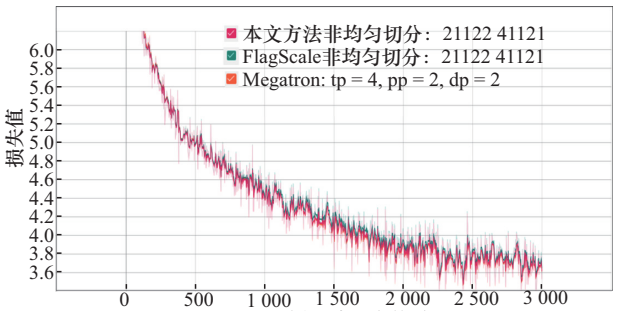
本文实验采用开源框架 Megatron 作为参考基准，开源异构训练框架 FlagScale 作为混训能力比较系统，在对齐并行策略后，与 FlagScale 比较不同训练迭代下相对于 Megatron 的损失误差值。

实验对照本文混训系统与 FlagScale 的非均匀切分策略为 21122、41121（并行策略排列方式为 TP CP EP DP PP）；同时分别对照 Megatron 均匀切分比例 21124、41122，流水线并行维度中两类 AI 加速器分配到的模型总层数为 12 层、4 层，其中第一个 Megatron 基准流水线并行配置为 2 6 6 2 2 2（配置顺序为 stage1 layer1_1 layer1_2 stage2 layer2_1 layer2_2），第二个 Megatron 基准流水线并行配置为 1 12 1 4（配置顺序同上）。针对各训练迭代分别计算两种异构训练系统与 Megatron 基准值的绝对误差及相对误差，输出损失误差平均值、最大值。

本文方法、FlagScale 与业界基准收敛曲线比较如图 25 所示，本文方法、FlagScale 与业界基准收敛精度比较如表 1 所示，本文方法与 FlagScale 收敛精度比较如表 2 所示。根据实验结果，本文系统与



(a) Megtaron基准切分比例为21124



(b) Megatron基准切分比例为41122

图25　本文方法、FlagScale 与业界基准收敛曲线比较

表 1　本文方法、FlagScale 与业界基准收敛精度比较

损失误差指标	本文混训系统与 Megatron 基准值 1	FlagScale 与 Megatron 基准值 1	本文混训系统与 Megatron 基准值 2	FlagScale 与 Megatron 基准值 2
平均绝对误差	0.323 0	0.599 1	0.256 9	0.853 1
平均相对误差	0.015 1%	0.024 9%	0.011 0%	0.034 2%
最大误差	1.359 5%	2.617 5%	0.999 3%	3.533 8%

Megatron 基准值的平均绝对误差范围为 0.256 9~0.323 0、平均相对误差范围为 0.011%~0.015 1%，最大相对误差不超过 1.36%；FlagScale 与 Megatron 基准值的平均绝对误差范围为 0.599 1~0.853 1，平均相对误差范围为 0.024 9%~0.034 2%，最大相对误差不超过 3.6%。从各项误差指标来看，本文方法损失误差均小于 FlagScale，表明本文方法收敛精度与业界基准更为接近，非均匀切分算法可以对齐均匀切分收敛精度。

表 2　本文方法与 FlagScale 收敛精度比较

损失误差指标	本文混训系统与 FlagScale
平均绝对误差	0.703 8
平均相对误差	0.028 8%
最大误差	2.452 5%

同时，为比较本文方法与 FlagScale 的收敛精度，表 2 计算了 2 种异构训练系统的损失误差值，平均绝对误差为 0.703 8、平均相对误差为 0.028 8%，最大相对误差为 2.452 5%。从整体来看，本文方法具有更好的收敛特性，在精度和收敛速度上均具有更大的优势。

3.2.7　训练性能

为评估模型训练性能，采用损失曲线、LLM 语言模型困惑度（PPL, perplexity）作为评价指标。其中，损失曲线用于验证模型在混合训练条件下能否正常收敛，PPL 用于验证模型是否有较好的语言数据预测能力，PPL 值越低模型预测能力越好。

为体现模型收敛趋势，通过每隔 N 次迭代记录损失值进行损失曲线绘制；为体现模型预测能力，采用 PPL 曲线反映语言模型性能结果，实验过程将数据集按照特定比例分割为训练集、验证集、测试集，各执行多次迭代进行模型评估，绘制 PPL 曲线。

NV H100+BI-V150 的损失曲线及 PPL 曲线如图 26 所示。由图 26 可以看出，当混合集群为 NV H100+BI-V150 时，LLaMA2 13B 模型训练损失曲线呈下降趋势，可正常收敛，且 PPL 曲线呈下降趋势；NV GPU 与天数智芯 GPU 合池对模型性能没有影响。NV H100+壁砺 106B 的损失曲线及 PPL 曲线如图 27 所示。当混合集群为 NV H100+壁砺 106B 时，LLaMA2 13B 模型训练损失曲线呈下降趋势，可正常收敛，且 PPL 曲线呈下降趋势；NV GPU 与壁仞 GPU 合池对模型性能没有影响。

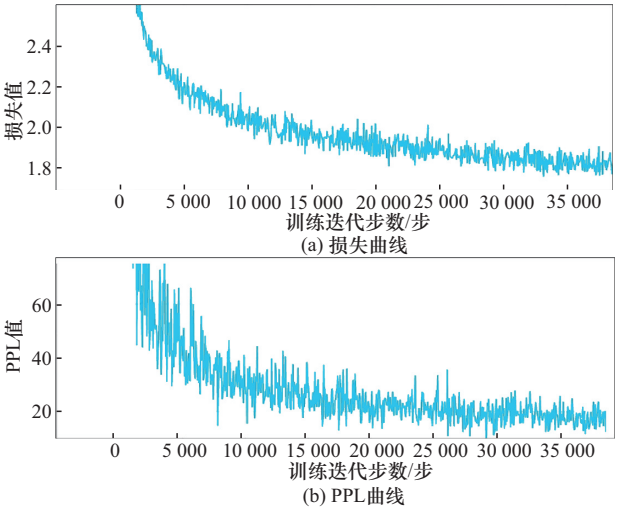


图 26　NV H100+BI-V150 的损失曲线及 PPL 曲线

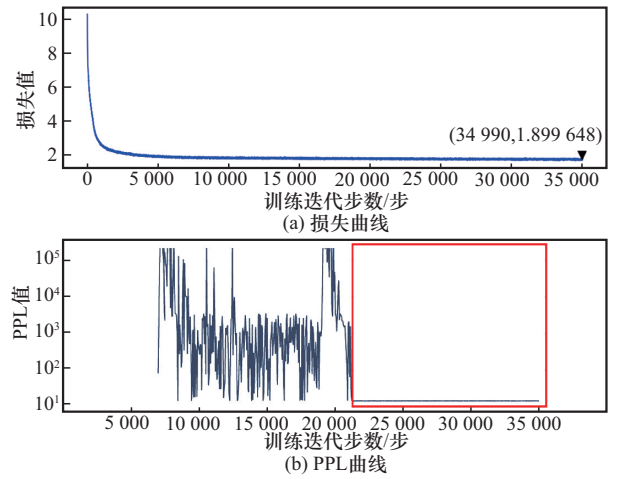


图 27　NV H100+壁砺 106B 的 损失曲线及 PPL 曲线

4　结束语

本文提出了一种面向异构混合算力的分布式并行训练技术，该技术可通过非均匀计算任务切分算法、非均匀切分性能预测技术、混训通信技术实现不同类型 AI 加速器在单一大模型任务下的混合分布式训练，该技术在 Nvidia GPU、天数智芯、壁仞混合算力资源池进行了可行性验证，经分析异构混合加速比均超 90%，且模型训练损失曲线及 PPL 曲线均呈下降趋势，异构多芯混合训练效率损耗较低，可满足大模型训练需求，异构芯片混池训练对模型性能没有影响。

未来，将持续深入探索智算异构混合并行训练机制，逐步拓展验证方案及模型场景，攻破大模型混合训练系列挑战，相关工作可考虑包括。

1) 目前异构非均匀切分性能预测技术主要考虑

在流水线并行策略场景下进行设计，后续将进一步引入数据并行、张量并行等其他并行策略场景迭代设计优化，并进一步提升性能预测结果的稳健性。

2) 需进一步探索本文方法对超大规模训练任务的适用性，验证本文方法在最新高端AI加速器、更大规模AI加速器集群、更大规模模型上的训练效果，并采用多维度评测数据集分析大模型应用本文方法的实际模型性能。

3) 需面向更大规模集群训练需求，在资源调度、管理编排等能力上进行技术探索，增强本文方法对超大规模训练的适用性及扩展性。

参考文献：

[1] BROWN T, MANN B, RYDER N,et al. Language models are few-shot learners[C]//NIPS'20: 34th International Conference on Neural Information Processing Systems. Michigan: Curran Associates Inc., 2020: 1877-1901.

[2] DEEPSEEK-AI, LIU A X, FENG B, et al. DeepSeek-V3 technical report[J]. arXiv Preprint, arXiv: 2412.19437, 2024.

[3] TOUVRON H, LAVRIL T, IZACARD G, et al. LLaMA: open and efficient foundation language models[J]. arXiv Preprint, arXiv: 2302.13971, 2023.

[4] LI S, ZHAO Y L, VARMA R, et al. PyTorch distributed: experiences on accelerating data parallel training[J]. arXiv Preprint, arXiv: 2006.15704, 2020.

[5] SHOEYBI M, PATWARY M, PURI R, et al. Megatron-LM: training multi-billion parameter language models using model parallelism[J]. arXiv Preprint, arXiv: 1909.08053, 2019.

[6] HUANG Y P, CHENG Y L, BAPNA A, et al. GPipe: efficient training of giant neural networks using pipeline parallelism[J]. arXiv Preprint, arXiv: 1811.06965, 2018.

[7] RAJBHANDARI S, RASLEY J, RUWASE O, et al. ZeRO: memory optimizations toward training trillion parameter models[J]. arXiv Preprint, arXiv: 1910.02054, 2019.

[8] NARAYANAN D, HARLAP A, PHANISHAYEE A, et al. PipeDream: generalized pipeline parallelism for DNN training[C]//Proceedings of the 27th ACM Symposium on Operating Systems Principles. New York: ACM Press, 2019: 1-15.

[9] KAPLAN J, MCCANDLISH S, HENIGHAN T, et al. Scaling laws for neural language models[J]. arXiv Preprint, arXiv: 2001.08361, 2020.

[10] DUAN J F, LI X H, XU P, et al. Proteus: simulating the performance of distributed DNN training[J]. arXiv Preprint, arXiv: 2306.02267, 2023.

[11] SCHAARSCHMIDT M, GREWE D, VYTINIOTIS D, et al. Automap: towards ergonomic automated parallelism for ML models[J]. arXiv Preprint, arXiv: 2112.02958, 2021.

[12] ZHANG S W, DIAO L S, WU C, et al. HAP: SPMD DNN training on heterogeneous GPU clusters with automated program synthesis[C]// Proceedings of the Nineteenth European Conference on Computer Systems. New York: ACM Press, 2024: 524-541.

[13] LI D C, WANG H Y, XING E, et al. AMP: automatically finding model parallel strategies with heterogeneity awareness[J]. arXiv Preprint, arXiv: 2210.07297, 2022.

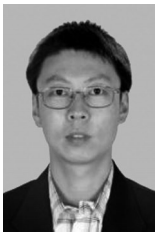
[14] CHO M, FINKLER U, SERRANO M, et al. BlueConnect: decomposing all-reduce for deep learning on heterogeneous network hierarchy[J]. IBM Journal of Research and Development, 2019, 63(6): 1-11.

[作者简介]

黄蕾（1991– ），女，福建永安人，中国移动研究院助理工程师，主要研究方向为机器学习系统、分布式并行计算、深度学习加速器等。

王升（1987– ），男，河南淇县人，中国移动研究院高级工程师，主要研究方向为NFV/SDN、异构计算、算力网络等。

班有容（1990– ），女，河北唐山人，中国移动研究院工程师，主要研究方向为异构训练、异构计算、云计算。



张昊（1979– ），男，山西晋中人，博士，中国移动研究院正高级工程师，主要研究方向为5G、云计算、新型智算、算力网络等。

张晓光（1980– ），男，河北张北人，中国移动研究院高级工程师，主要研究方向为智算中心软硬件架构设计、云计算自动化集成体系研究及系统构建等。

狄新凯（1992– ），男，山东枣庄人，博士，中国移动研究院工程师，主要研究方向为大模型分布式训练、集合通信库、编译系统。

许思（1994– ），男，河北石家庄人，上海无问芯穹智能科技有限公司研发工程师，主要研究方向为机器学习系统、分布式并行计算、深度学习加速器。

黄子潇（2002– ），男，北京人，上海无问芯穹智能科技有限公司实习生，主要研究方向为计算机系统结构。