DOI: 10.11992/tis.202108010

面向机器学习的分布式并行计算关键技术及应用

曹嵘晖[1,2], 唐卓[1,2], 左知微[1,2], 张学东[1,2]

(1. 湖南大学 信息科学与工程学院, 湖南 长沙 410082; 2. 国家超级计算长沙中心, 湖南 长沙 410082)

摘　要: 当前机器学习等算法的计算、迭代过程日趋复杂, 充足的算力是保障人工智能应用落地效果的关键。本文首先提出一种适应倾斜数据的分布式异构环境下的任务时空调度算法, 有效提升机器学习模型训练等任务的平均效率; 其次, 提出分布式异构环境下高效的资源管理系统与节能调度算法, 实现分布式异构环境下基于动态预测的跨域计算资源迁移及电压/频率的动态调节, 节省了系统的整体能耗; 然后构建了适应于机器学习/深度学习算法迭代的分布式异构优化环境, 提出了面向机器学习/图迭代算法的分布式并行优化基本方法。最后, 本文研发了面向领域应用的智能分析系统, 并在制造、交通、教育、医疗等领域推广应用, 解决了在高效数据采集、存储、清洗、融合与智能分析等过程中普遍存在的性能瓶颈问题。

关键词: 机器学习; 分布式计算; 倾斜数据; 任务时空调度; 资源管理; 节能调度; 跨域资源迁移; 并行优化; 图迭代算法; 智能分析系统

中图分类号: TP18　　**文献标志码:** A　　**文章编号:** 1673−4785(2021)05−0919−12

中文引用格式: 曹嵘晖, 唐卓, 左知微, 等. 面向机器学习的分布式并行计算关键技术及应用 [J]. 智能系统学报, 2021, 16(5): 919–930.

英文引用格式: CAO Ronghui, TANG Zhuo, ZUO Zhiwei, et al. Key technologies and applications of distributed parallel computing for machine learning[J]. CAAI transactions on intelligent systems, 2021, 16(5): 919–930.

Key technologies and applications of distributed parallel computing for machine learning

CAO Ronghui[1,2], TANG Zhuo[1,2], ZUO Zhiwei[1,2], ZHANG Xuedong[1,2]

(1. College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China; 2. National Supercomputer Center in Changsha, Changsha 410082, China)

Abstract: At present, the calculation and iteration process of algorithms such as machine learning is becoming more and more complex. Sufficient computational power is the key to ensure the landing effect of artificial intelligence application. In view of this, this paper first puts forward a task space-time scheduling algorithm adapted to the distributed heterogeneous environment of skew data, which effectively improves the average efficiency of tasks such as machine learning model training. Then, the high-efficiency resource management system and energy-saving scheduling algorithm in distributed heterogeneous environment are proposed to realize the dynamic prediction based cross-domain computing resource migration and voltage/frequency dynamic regulation in distributed heterogeneous environment, which saves the overall energy consumption of the system, and then, the distributed heterogeneous optimization environment adapted to the iteration of machine learning/deep learning algorithm is constructed, and the basic method of distributed parallel optimization for machine learning/graph iteration algorithm is proposed. Finally, the intelligent analysis system for field-oriented applications is researched and developed, and popularized in manufacturing, transportation, education, medical and other fields, which solves the performance bottleneck problems that are common in the process of high-efficiency data collection, storage, cleaning, fusion and intelligent analysis.

Keywords: machine learning; distributed computing; skew data; task space-time scheduling; resource management; energy-saving scheduling; cross-domain resource migration; parallel optimization; graph iteration algorithm; intelligent analysis system

收稿日期: 2021−08−11.

基金项目: 国家重点研发计划项目(2018YFB1701400); 国家自然科学基金项目(92055213, 61873090, L1924056, 62002114); 金融及产业数据驱动下的智慧园区云平台研发及产业化项目(XMHT20190205007); 广东省重点领域研发计划项目(XMHT20190205007)深圳市科技计划项目(JSGG20180507183023239).

通信作者: 唐卓. E-mail: ztang@hnu.edu.cn.

以超级计算、云计算为计算基础设施, 以大数据分析、从海量经验数据中产生智能的人工智能 2.0 时代的浪潮正在袭来[1-2]。互联网、人工智能应用的蓬勃发展, 在海量数据的处理分析上面临巨大的挑战: 传统数据平台的并行计算能力、

弹性存储能力以及智能化数据分析能力难以满足各行业海量数据在采集、存储和分析上对计算资源的迫切需求[3-6]。数据驱动的人工智能技术飞速发展，给互联网、智能制造、智慧城市等应用领域在数据采集、处理和分析框架上带来了巨大的机会[7-9]。

与此同时，近年来蓬勃发展的企业应用、互联网应用在海量数据的处理分析上也面临巨大的挑战：传统数据平台的并行计算能力[10]、弹性存储能力以及智能化数据分析能力难以满足行业海量数据的采集[11]、存储和分析的需求[12]。

而目前国内人工智能行业、大数据行业发展的主要矛盾是：大多数企业看得到数据，但对数据如何采集[13]、存储[14]、分析[15]、提供智能决策等方面缺乏成熟有效的平台支撑，技术准入门槛高[16-17]。

1）流数据、非结构化数据的处理和分析往往需要动态可扩展的计算和存储能力，传统的以服务器集群、SQL 数据库为主流架构的企业数据中心基础设施无论在硬件和软件容量上都不具备实时扩展的能力，很难满足企业数据处理应用对资源的弹性需求[18-19]。

2）现有的面向非结构化的数据存储架构基本上是基于 NoSQL 分布式文件系统，这给传统的以 SQL 数据库编程为主要技能的程序员带来了困扰[20-21]。

3）现有的传统企业基于数据库的分析和处理的应用往往不具备按照数据分块进行并行处理的能力。而现有主流并行编程框架对于一般的企业开发人员来说又难以短时间掌握。这使得以 Hadoop/Spark、Flink 等为代表的大数据并行存储和处理框架的应用很难得到较大面积的推广和应用[22-24]。

4）以人工智能经典算法、机器学习模型为核心的数据挖掘框架是目前进行大数据分析的主要手段。但对于传统企业的开发人员来说，同样面临着人工智能算法门槛太高，难于掌握的困境，使得一般的软件公司很难组建面向行业数据分析处理和挖掘的研发团队[25-26]。

课题组依托的国家超算长沙中心，作为我国在云计算、大数据及行业应用的重大战略基础设施，其核心设备天河一号超级计算机与云服务器集群具备 PB 级的数据存储、并行处理和分析挖掘的能力，能有效解决传统企业在面向海量数据处理中所遇到的计算、存储和算法瓶颈。

面向我国行业领域对大数据并行处理与智能分析技术和服务能力提出的迫切需求，本文提出了高效能数据并行处理与智能分析系统，为相关行业提供数据存储、分析和挖掘的智能化云服务，有效降低传统企业基于超级计算机、云服务集群等来实现大数据智能分析的使用门槛。该系统有效地突破了数据采集、存储、压缩、分析、挖掘过程中在数据并行处理体系结构、人工智能算法、并行编程模型方面存在的技术瓶颈，一方面有效发挥了课题组所依托的国家超级计算长沙中心作为高性能数据处理基础设施的公共服务能力，另一方面将为领域企业提供了行业数据并行处理与智能分析的能力，提升了我国相关骨干企业的创新能力。

1　研究方案

本文的研究应用方案如图 1 所示。

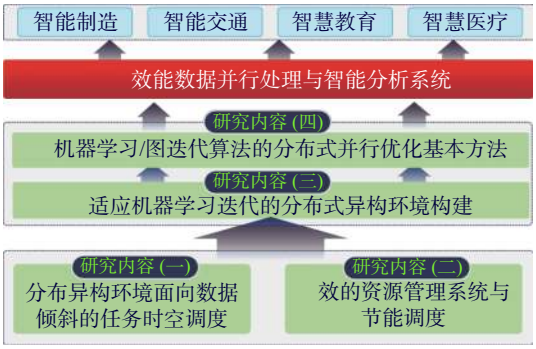

图 1　本文研究总体框架

Fig. 1　General introduction of the research

研究应用方案具体包括：

1）首先针对大多数云环境中服务器内存资源平均使用率过低问题，提出了基于服务器内存预测的虚拟机动态预测部署及任务节能调度模型。在此基础上，针对 Hadoop/Spark 的数据处理过程，设计并实现了一种面向倾斜数据 Shuffle 过程的任务调度策略：一方面通过 Reduce 任务放置策略减少 Spark/Hadoop 集群的内部通信量，通过 Reducer 放置算法来实现任务本地化，以减少系统的中间数据传输量。

2）提出和研发了分布式异构环境下高效的资源管理系统与节能调度算法，针对各种迁移模型的场景，适配性能最优的计算资源迁移模型，并基于 OpenStack 云平台实现了面向数据中心集群的跨域计算资源迁移基础设施，能兼容多数云平台/数据中心虚拟机迁移算法，并支持目前流行的 Ceph、KVM(kernel-based virtual machine) 等存储和计算框架，实现了支持计算资源、存储资源调度算法的独立封装和部署的多数据中心资源管理体系结构。在此基础上，针对当前云环境中服务器内存资源平均使用率过低问题，提出了一种基于

服务器内存预测的分配机制下的虚拟机动态预测部署模型 VM-DFS(virtual machine dynamic forecast scheduling)。同时针对虚拟机动态迁移问题，提出了一种基于动态预测的虚拟机迁移模型 VM-DFM(virtual machine dynamic forecast migration)，解决了动态迁移过程中，如何从服务器上选择合适的虚拟机进行动态迁移，从而达到整体节能的目标。

3）海量数据存储和高并发用户访问需要分布式环境，但以异构众核等为主要计算部件的参数训练过程无法适应分布式系统。原生的 Spark/Flink 等分布式数据处理框架也无法高效适用于深度学习的参数训练，GPU 等高性能计算单元又无法应对海量数据的分布存储和计算，且难以支撑高并发的数据访问。因此，本文针对深度学习增量迭代的运算过程，研究迭代过程中的中间共享结果在 GPU 内存及 Cache 内的存储和管理以及线程间的共享访问机制。针对现有流行的分布式大数据处理框架，研究其在 CPU/GPU 异构环境中的体系结构扩展优化模型，突破 Spark RDD 等在 GPU 环境中的数据结构和体系结构的重新设计，研究增量迭代过程中计算结果在 GPU 线程间以及 Spark 进程间的共享模型，实现其在异构计算环境下的缓存和持久化。

4）本文针对 DNN(deep neural networks)、CNN(convolutional neural networks)、RNN(recurrent neural network) 等典型深度学习模型训练中的参数迭代过程进行了深入研究，总结出增量迭代发生的模型、数据特征，发现了其训练过程可以实行增量迭代优化的条件和时机，提出了普适性的深度学习增量迭代优化方法；针对现有 Spark/Flink 分布式大数据处理框架，提出了其在 CPU/GPU 异构环境中的体系结构扩展优化模型，设计并实现了一种在 Spark/Flink 计算容器与 GPU 核心间的高效通信方式，将传统分布式深度学习框架的运行效率提升数倍。在此基础上，提出了分布式环境中的并行条件随机场模型，将训练效率提升了 3.125 倍；提出了一种并行维特比算法，减少了计算步骤之间存在冗余的磁盘读写开销和多次资源申请的问题，加速比达到 6.5 倍。

2　分布异构环境面向数据倾斜的任务时空调度

倾斜是自然界与人类社会中数据属性客观存在，会造成集群计算节点负载不均衡、排队现象/空等待现象普遍存在，集群内部吞吐率低下，大幅度降低了系统的实际应用效率[27-30]。鉴于此，本文研制了分布异构环境面向数据倾斜的任务时空调度策略，本地化任务放置算法，以及分布式并行处理框架中的内部数据均匀分片方法。形成了面向机器学习训练任务的任务调度理论与方法。

2.1　基于 Spark 平台的中间数据负载平衡设计

自然界中数据分布多数在理论上都是倾斜的，导致倾斜的原因复杂且无法避免，因此在处理数据时，如果没有精心设计数据划分或任务调度会极大程度地造成计算资源的浪费和系统整体性能偏差。由此可知，数据偏斜带来的负载均衡问题是分布式计算平台中优化的难点和重点[31-33]。对于集群系统，数据对应任务，数据偏斜带来的任务负载均衡问题会导致分布式系统的资源利用率低、计算执行时间长且能耗高。本文基于现有的分布式计算框架 Spark，优化 Spark 计算框架下 shuffle 执行过程中 bucket 容器中的数据偏斜导致的负载不平衡问题。本文提出了一种面向中间偏斜数据块的重新划分和再合并算法，通过两个重要操作以缓解 shuffle 操作后 reduce 任务中的负载不平衡问题。图 2 是 SCID 系统架构模块，该模块包含系统中任务执行的流程和 shuffle 过程。

在这种分布式集群体系架构中，每一个小块的分片数据是文件的组织单位，该分片在 HDFS (hadoop distributed file system) 中是默认的固定大小。在执行一个 map 任务时，客户端的初始数据首先被加载到分布式文件系统 (HDFS) 中，每个文件由多个大小相同的数据块组成，称为输入分区。每个输入分区都被映射为一个 map 任务。在本文中，使用 $I \subseteq K \times V$ 来代表 m 个 map 任务的中间结果，K 和 V 分别代表键和值的集合。一个 cluster 是某一个 key 值对应的<键,值>对的集合，其一个子集为

$$C_k = (k,v) \in I, k \in K, v \in V \tag{1}$$

在图 2 中使用分区函数 Π 决定一个中间元组的分区号：

$$\Pi : K \to \{1,2,\cdots,p\} \tag{2}$$

因此，shuffle 过程中 map 端输出的中间结果被划分为 p 个大小不同的分区，分区号根据元组的键值通过 hash 计算得到。因此所有 key 相同的元组都会被指向相同的分区，因为它们都属于一个 cluster。分区是一个包含一个或多个 clusters 的容器。因此，定义一个分区为

$$P(j) = \bigcup_{k \in K:\Pi(k)=j} C(k) \tag{3}$$

基于以上定义，本文提出了一种新颖的 Spark 作业负载均衡方法，设计了一个负载均衡模块来重新划分使之实现任务的均衡划分。该模块的执

行流程如下：在 Spark 提交作业后，负载均衡器启动并分析作业特点给出均衡分区策略。该策略在 Spark 作业 shuffle 阶段指导系统对中间结果数据进行分割和重组，重组结果 clusters 到一个或多个 buckers 之中，从而实现均衡分区。本文提出的负载均衡模块在 Spark 基础上设计，主要包括两个重要过程，分别为数据的采样和 cluster 的分割组合，其中在数据抽样阶段，重点的是对 clusters 大小的进行预测。图 3 代表了一种改进的工作流的 Spark 作业，其中的一个核心组件是负载均衡模块。

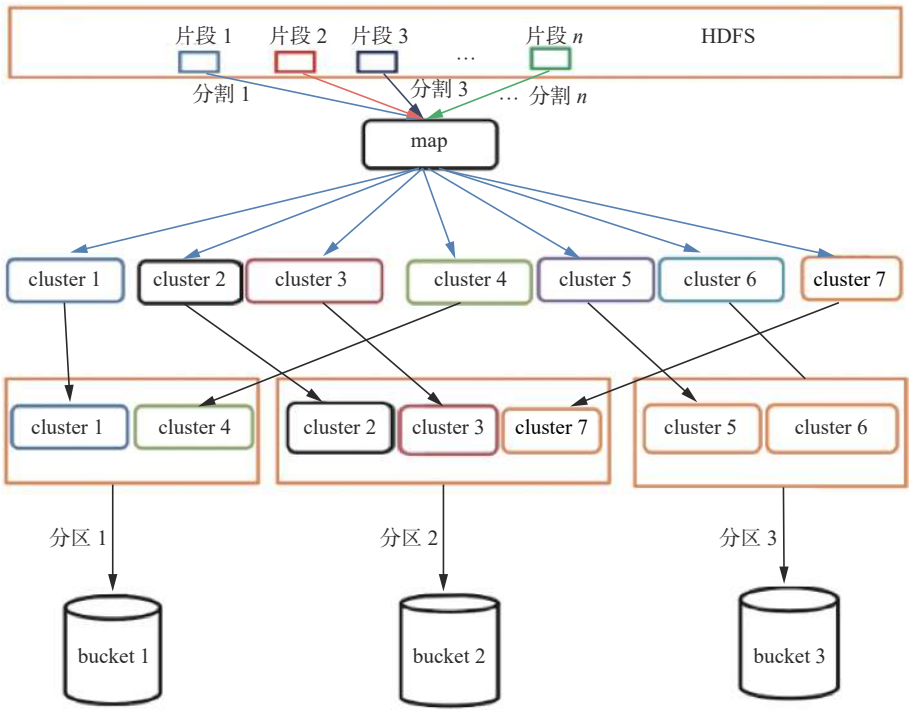

图 2　Spark 中 shuffle 数据分布过程

Fig. 2　Process of shuffle data distribution in Spark

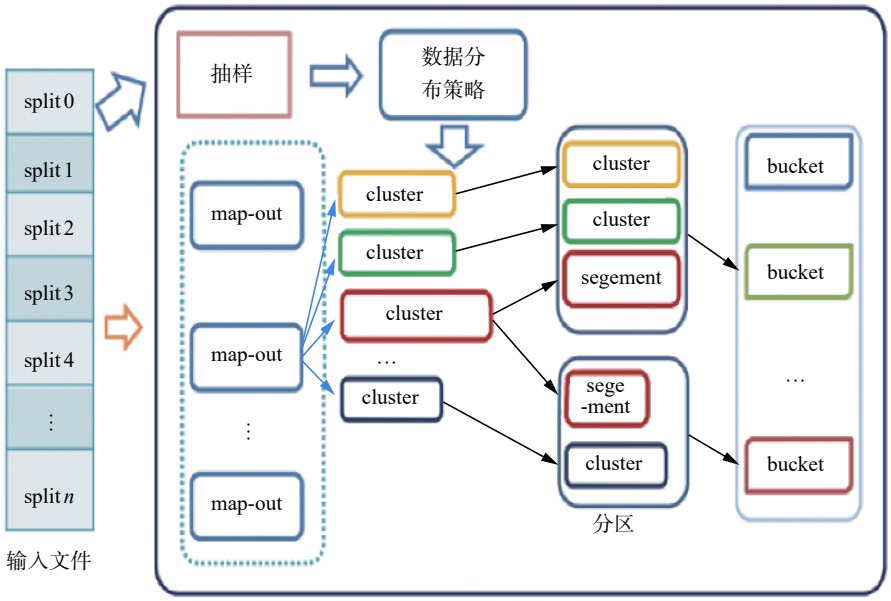

图 3　架构与负载均衡

Fig. 3　Architecture and load balancing

在 cluster 分割重组的过程中，第一要义是分割以 bucket 的大小作为目标进行分割，特别是对于一些超大的 clusters 应该尽量分成多个 bucket 大小的数据块，方便重组填充的过程。众所周知，现有的分布式大数据处理平台如 Hadoop/Spark 体系架构中在数据处理阶段缺乏对计算数据的真

实分布的清晰认知[34]，抽样数据虽然不能保证真实地反映全体数据的分布特征，但基于其结果来近似估计数据的整体分布也可以实现较好的结果。在此基础上，本文提出了一种改进分局均衡策略来缓解现有分布式并行计算框架中的数据偏斜问题。

2.2　面向分布式处理的抗数据倾斜分片机制

随着大数据时代的到来，信息爆炸使得数据的规模和复杂度都在增长，大数据并行计算中数据偏斜问题也日趋严重，成为一个亟需解决的问题。目前，大数据处理主流框架中对抗数据偏斜的能力都普遍较弱[35-37]。普适性的分布式并行计算框架中通常假设数据在计算过程中是均匀分布的，这跟现实数据的分布特征背向而驰。严重的数据偏斜程度会使集群计算系统的计算能力直线下降，引发资源利用率低和任务执行过慢等问题。鉴于此，本文提出了一种密钥重分配和分裂分区算法（SKRSP）来解决分区倾斜，该算法同时考虑了中间数据的分区平衡和 shuffle 算子后的分区平衡。SKRSP 策略的整体架构如图 4 所示。

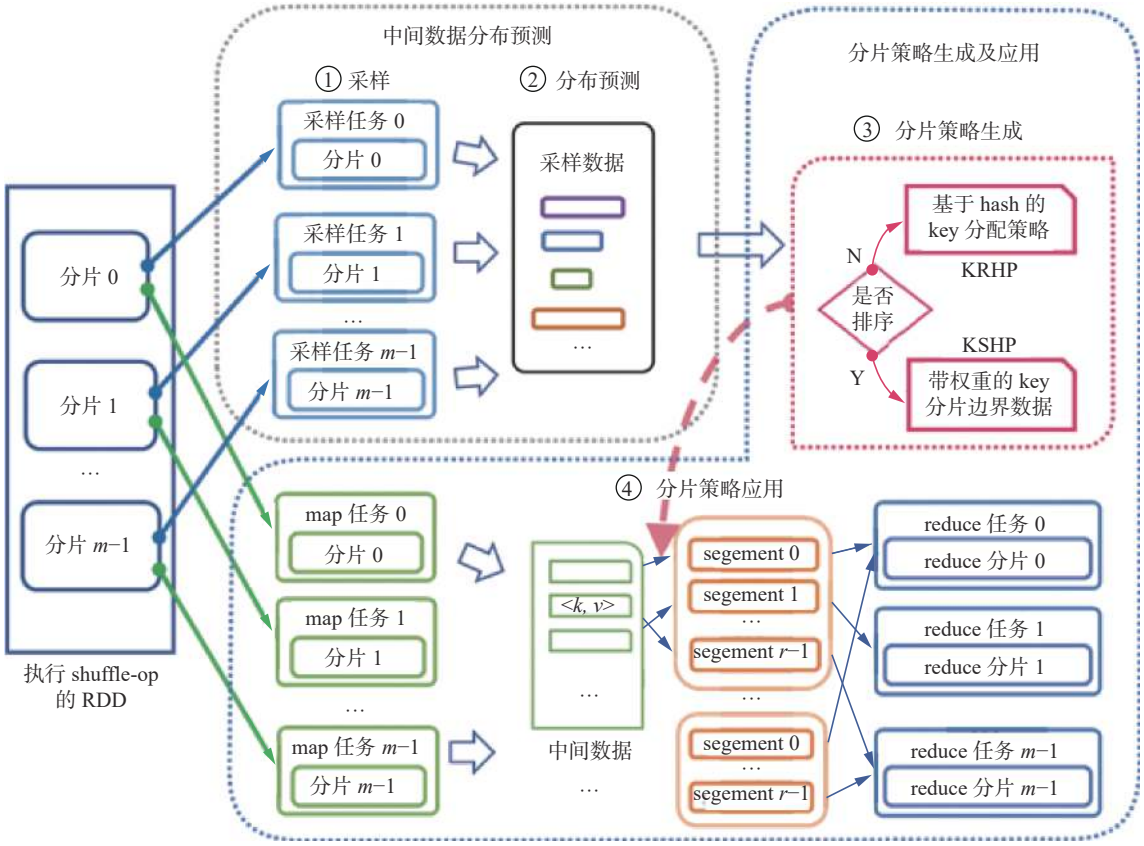

图 4　SKRSP 整体架构

Fig. 4　General introduction to SKRSP

SKRSP 整体框架包含了两个主要部分：中间数据分布预测、分片策略的生成与应用。

1）为了避免 reduce 任务之间的数据偏斜，需要在 shuffle 阶段之前估计中间数据的 key 分布。因此，必须在常规作业之前输入地图任务时启动先前的示例作业。本文在不同的分区上并行实现了基于步骤的拒绝采样算法。所有的样本和对应的采样率都是从不同的 map splits 中收集的，它们构成了通过采样率计算每个 key 的权重的输入。在此基础上，可以估计中间数据的一般 key 分布。

2）分片策略的生成与应用。本系统根据 Spark 作业的具体应用场景，采用不同的方法生成分配策略。对于这些属于排序类的应用程序，提出了 KSRP 算法来确定加权边界。最终的 key 重新分配策略可以通过其他 KRHP 算法获得。具体来说，采样中间数据 key 的分布是系统用于决策分区策略的依据。一方面，如果操作结果无需排序，基于 hash 的 key cluster 分片方法将被采用；另一方面，如果操作结果是需要进行排序，基于 range 的 key cluster 分片策略将被采用。因此，就得到了不同的分片策略。在 shuffle 写数据的阶段，在上一个步骤中获得的分片策略会指导每个<key, value>对其进行分区计算，从而获得其 reduce 端的分区 ID 号。该 ID 号就是每个 map 任务计算后

的中间输出结果，需要写到磁盘的顺序位置。最终这些中间结果生成一个数据文件和索引文件。在数据文件中，一个数据段（segement）是一块索引号相同的区域。接下来进入 shuffle 的读阶段，每个 reduce 任务将从各个 map 任务执行的节点上根据索引文件拉取数据。也就是说，具有相同 reduce 索引号的键值对数据组成一个 reduce 分区，将被一个对应的 reduce 任务处理。经过这样的过程，上一步生成的分片策略便应用到了 Shuffle 过程中实际的数据划分中来。

在实际的 Spark 集群上对 SKRSP 算法进行了评估，并与其他算法进行了对比实验如表 1。在采样率为 3.3% 的情况下，SKRSP 算法明显优于其他采样方法，且误差小于 LIBRA，仅为 70。

表 1　采样精确度实验结果

Table 1　Experimental results of sampling accuracy

采样方法	采样率/%	均匀分区5 000 keys	均匀分区50 000 keys	倾斜分区5 000 keys	倾斜分50 000 keys
SKRSP	3.30	1 311	296	581	207
Range采样器	3.30	1 622	460	883	245
随机采样器	3.30	1 897	361	743	218
SKRSP	20	288	110	232	78

3　分布式异构环境下高效的资源管理系统与节能调度

3.1　分布式异构环境下的计算资源跨域迁移

数据中心等分布式异构基础设施已经成为现代各行各业的基础建设，从为中小型公司提供业务支撑数据机房，到大型 IT 公司的 IDC(internet data center)[38-39]。然而服务中断、资源属性等特性对资源跨域迁移的需求越来越大。结合项目组提出的多云资源级联平台，本文基于 OpenStack 实现了一个面向数据中心集群的跨域计算资源迁移基础设施，实现了多云环境下 VM(virtual machine)跨域迁移，有效地满足一种或多种用户、资源需求，并在此基础上实现了支持计算资源、存储资源调度算法的独立封装和部署的多数据中心资源管理体系结构。该结构如图 5 所示。

如图 5 所示，如若需要将 VM 从 Pod 1 迁移到多元环境下的 Pod 2 下，首先 Pod 1 的计算组件 Nova 需要向顶层 OpenStack 云平台发送迁移消息，顶层 OpenStack 收到该消息后交予 Nova API-GW 处理，并发送给 MSG.Bus，为发送给 Cascading Manager 其他模块做准备。随后，Nova API-GW 通过消息队列将该迁移信息发送给数据库，请求修改资源路由表中相关资源信息。同时，也通过异步作业机制给迁移的目的云实例发送迁移消息。VM 迁移的目的云实例接到该请求后发送给 Pod 2 的计算组件 Nova 。在多云架构顶层为迁移做资源管理信息修改时，底层的两个云实例之间完成虚拟机冷迁移所需镜像文件和内存数据的传输。

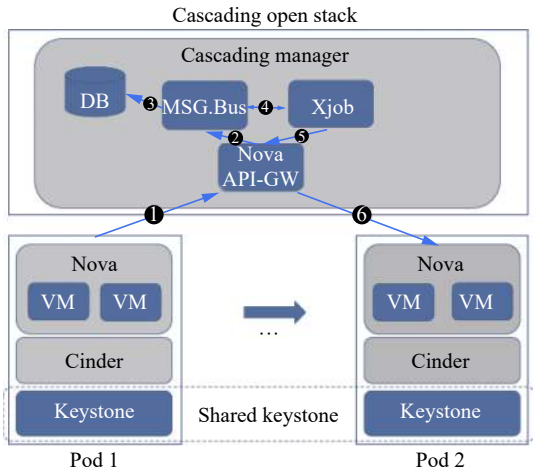

图 5　跨域 VM 迁移机制

Fig. 5　Cross-domain VM migration mechanism

该架构在真实多云环境下进行了实验，实验结果证明了该架构的有效性和高效性，提升跨域资源的使用效率。此外，在多云平台上的用户使用过程中，该架构也能有效地降低因虚拟机突发迁移带来的用户宕机体验率。

3.2　基于服务器内存预测的虚拟机分配机制

通过对云环境下虚拟机部署方式的研究，针对现有云服务器中的内存使用率低导致各类资源平均使用率过低[40-41]，本文提出一种新型虚拟机部署机制 VM-DFS，基于云服务器内存预测下的虚拟机动态部署模型。该模型考虑虚拟机运行过程对云服务器内存消耗的动态变化，结合虚拟机部署已有的研究方案，将部署过程构建为某一类装箱模型，在此基础上，再结合 FFD(first-fit decreasing)算法对虚拟机部署算法近似求解；与此同时，虚拟机部署过程中结合内容等资源的预测机制，通

过对各个虚拟机历史内存消耗数据的统计分析，使用基于时间序列的自回归二阶模型进行内存动态预测。在满足各个虚拟机对内存 SLA (service level agreement) 要求的前提下减少服务器的启动数量。并对每个服务器的内存分配设置一个阈值 L_m，设置平衡因子 r 作为超过阈值的过载比例。实验结果显示，VM-DFS 算法能够在满足 SLA 要求的前提下，提高服务器内存资源使用率。

在此基础上，为确保云环境中内存资源的 Qos 要求，当物理服务器内存消耗 r 值时，需要进行虚拟机的动态迁移。鉴于此，本文提出一种新型虚拟机动态迁移模型 (virtual machine dynamic forecast migration, VM-DFM)，该算法解决了在虚拟机的动态迁移过程中，如何从“热点”服务器上待迁移虚拟机列表中选择合适的虚拟机进行动态迁移。

4　适应于机器学习/深度学习算法迭代的分布式异构环境构建

针对机器学习/深度学习算法迭代过程中的算力、架构瓶颈及计算效率低等问题[42-43]。提出了普适性的深度学习增量迭代优化方法；针对现有 Spark/Flink 分布式大数据处理框架，在此基础上提出了其在 CPU/GPU 异构环境中的体系结构扩展优化模型，设计并实现了一种在 Spark/Flink 计算容器与 GPU 核心间的高效通信方式，解决了分布式异构环境中的计算效率问题。

4.1　机器学习/深度学习增量迭代优化方法

众所周知，算力一直以来是人工智能发展的最大瓶颈。以异构众核等高性能处理器为主要计算部件的机器学习/深度学习参数训练过程并不适用于分布式系统，传统的机器学习算法因其无法保证数据分片分开训练是否能与整体集中训练结果保持一致，需要在分布环境下进行并行优化与适应性改进。

鉴于此，本文针对分布式机器学习体系结构中的并行优化问题，提出了机器学习/深度学习增量迭代优化模型及其分布式异构 CPU/GPU 集群体系结构的优化设计方法。在此基础上，针对 DNN/CNN/RNN 等典型深度学习模型训练中的参数迭代过程，通过总结增量迭代发生的模型、数据特征，揭示了其训练过程可以实行增量迭代优化的条件和时机等客观规律，提出了普适性的深度学习增量迭代优化方法；提出并实现了一种在 Spark/Flink 计算容器与 GPU 核心间的高效通信方式，在兼具各个节点 GPU/MIC 众核计算能力的同时，利用分布式组件间的通信协议完成了各个服务器节点的协同运算。

4.2　分布式异构 CPU/GPU 集群体系结构的优化设计方法

考虑到目前 Spark 分布式框架无法有效利用计算节点上的多 GPU[42]，本文提出了 MGSpark 系统：一个 CPU-GPU 分布式异构环境下多 GPU 工作负载均衡的计算框架。MGSpark 系统能有效地将 GPUs 融入到 Spark 框架中，充分挖掘计算节点上的多 GPU 的计算能力，使集群中的 GPUs 工作负载达到均衡，如图 6 所示。

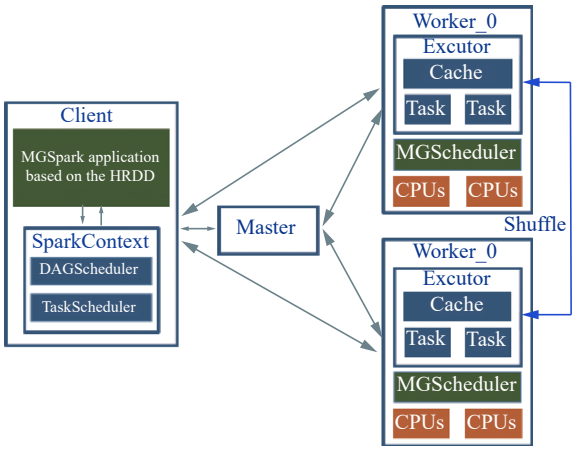

图 6　MGSpark 系统架构

Fig. 6　System architecture of MGSpark

本文建立了与原有 Spark RDD(resilient distributed datasets) 编程模型相兼容的 GPU 加速的编程模型，使编程人员创建 GPUs加速的 Spark 应用程序更加简便。为了优化主机端和设备端的数据通信，MGSpark 提出了一个多 GPU 环境下的异步 JVM-GPU 数据传输方案。

MGSpark 架构与 Spark 运行时相兼。因此 Spark 的任务调度和错误恢复机制被保留下来。Standalone 模式下的 MGSpark 系统框架如图 7 所示，保留着 Spark 运行时的所有组件（DAGScheduler、TaskScheduler、excutor）。作者还扩展了 RDD 模型来融合 GPU 和 Spark 的计算模型，以方便编程人员使用扩展的 RDD 编程模型来创建 MGSpark 应用程序，并使用 GPUs 进行加速。新增加的系统组件是 MGTaskScheduler，它驻留在每个 Worker 节点上。MGTaskScheduler 负责将 excutor 上的 Tasks 卸载到节点上的 GPUs 上执行，进行多 GPUs 工作负载均衡调度。

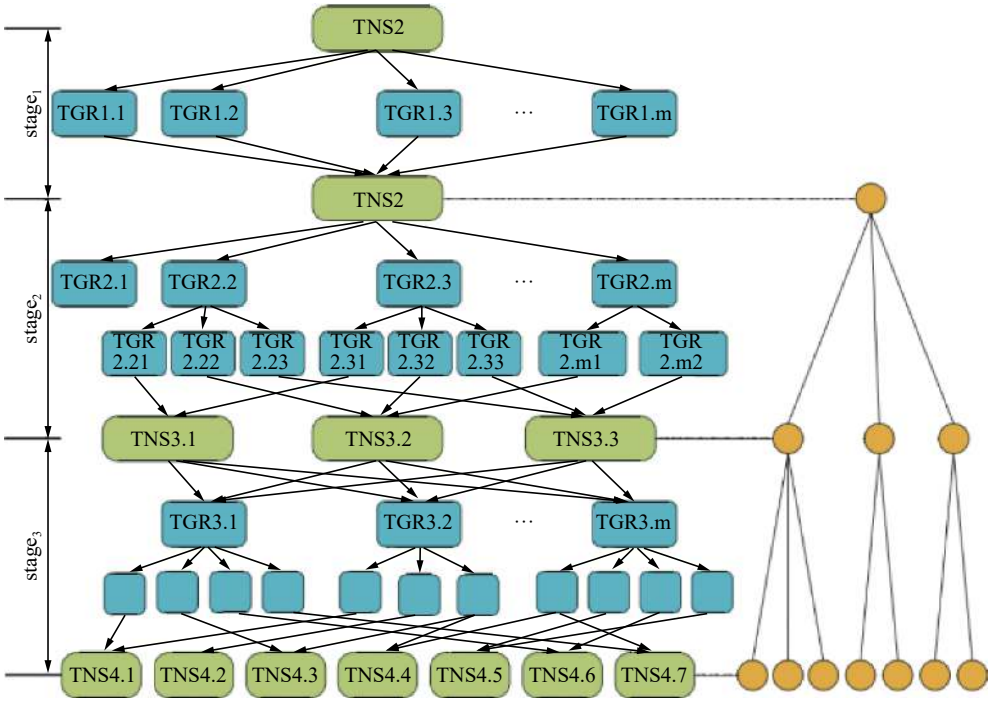

图 7　PRF 决策树模型训练过程的任务 DAG 模型

Fig. 7　Task DAG model of PRF decision tree model training process

使用扩展的 RDD 编程模型所创建的 MGSpark 应用程序在 Client 节点上被提交。Master 为应用程序分配所需的集群资源，主要包括内存和 CPU 资源。一个 DAG graph 根据 RDDs 之间的依赖关系被创建。DAG-Schedule 将 DAG 图划分为多个有先后顺序的 stage。每个 stage 划分为一系列可以并发的 Tasks 通过 Task-Scheduler。Task-Scheduler 根据集群每个节点资源状态调度 Tasks 到 workers 的进程上执行。与源生 Spark 框架不同（在 Spark 中 GPU 不能被识别和使用，Tasks 必须被调度到 CPU），MGspark Tasks 可以将计算与将要处理的数据卸载到 GPUs 上去进行加速通过 MGTaskScheduler 组件。

在此基础上，本文提出了基于 CUDA 流的异构任务执行模型 (MGMS)，可以充分平衡 GPUs 工作负载。并且将 MGMS 模型整合到最新版本的 Spark 分布式计算框架中开发了 MGSpark 计算框架。

Task 是 Spark 的最小调度和并发执行单元，每个 Task 需要顺序处理一个 Partition 的数据量。但是由于各个 Partition 之间的数据量不一样，特别是执行完 shuffle 类的算子，partition 之间的数据量差别更为明显。为了利用 GPUs 进行加速，将 Tasks 卸载到设备端形成 GTasks。如果将 GTask 作为一个最小执行单元分配设备资源：设备内存资源和 CUDA 流资源，调度到 GPUs 上去执行，会造成计算节点上各个 GPU 之间的工作负载不均衡。为了能平衡计算节点上各个 GPU 之间的工作负载，本文提出了一个任务分解执行模型。该模型主要包括两个部分：自动数据切片机制和自动任务分解机制。

5　面向机器学习/图迭代算法的分布式并行优化

针对机器学习/图迭代算法过程中的分布式并行优化中的计算效率等问题[43-44]。提出了面向机器学习算法的分布式并行优化模型、分布式环境中的并行条件随机场模型、并行维特比算法、基于冗余距离消除和极端点优化的数据聚类方法。解决了机器学习分布式优化的问题，突破了大规模高效能数据并行处理系统的算力瓶颈。

5.1　分布式环境中的并行条件随机场模型

条件随机场 (conditional random fields) 是一种概率图模型[45-46]。它是一种机器学习算法，需要多次迭代。条件随机场在标记或分析序列数据方面发挥了重要作用，并取得了显著的效果。条件随机场结合了最大熵模型和隐马尔可夫模型的特点，但隐马尔可夫模型不能直接看到其状态，不能应用复杂的特征。然而，根据这一思想，条件随机场模型可以很好地应用于依赖长距离和使用重叠特征的特征。同时，条件随机场可以解决其他判别模型中的标注偏差问题。为此，本文提出了一

种基于 Spark 的改进条件随机场模型 (SCRFs), 重点提高算法处理大数据的效率。该模型有以下创新: 为了加快速度, 将迭代过程中多次使用的中间数据缓存到内存中; 利用特征哈希的方法降低特征的维数; 对于梯度更新策略, 本文选择 Batch-SGD。基于上述创新, 可以有效地提高处理的时间效率。

参数估计是条件随机场模型中最重要的阶段。在处理大规模数据时, 模型的训练时间会大大增加, 需要花费大量的学习时间。大量实验表明, LBFGS 的第一步是训练过程中的主要环节。LBFGS 约 90% 的计算消耗处于第一步。如果能加快第一步, 整个训练过程的时间就会明显减少。因此, 条件随机场训练过程的并行化主要是并行计算目标梯度。

$$\frac{\partial L(\lambda)}{\partial \lambda_k}=\sum_{i=1}^{N}\left(\sum_{t=1}^{T} f_k(y_{t-1}^i,y_t^i,x_t^i)-\sum_{t=1}^{T}\sum_{y'y} f_k(y',y,x_t^i)\,p(y',y\,|\,x^i)\right)-\frac{\lambda_k}{\sigma^2} \tag{4}$$

通过式 (3) 可以得出第 1 部分是给定任意一个数据, 特征 f_k 的经验分布期望。可以描述为

$$E_{iD}[f_k]=\sum_{t=1}^{T} f_k(y_{t-1}^i,y_t^i,x_t^i) \tag{5}$$

第 2 部分是特征 f_k 的模型的期望分布:

$$E_{i\lambda}[f_k]=\sum_{t=1}^{T}\sum_{y'y} f_k(y',y,x_t^i)p(y',y\,|\,x^i) \tag{6}$$

经过简单的替换, 得到:

$$\frac{\partial L(\lambda)}{\partial \lambda_k}=\sum_{i=1}^{N}(E_{iD}[f_k]-E_{i\lambda}[f_k])-\frac{\lambda_k}{\sigma^2} \tag{7}$$

在求特征 f_k 的模型的期望分布的时候需要用到前面的 sum-product 信念传播算法, sum-product 能推断出模型的各边际分布概率。

$$E_{iD}[f_k]=\sum_{t=1}^{T} f_k\left(y_{t-1}^i,y_t^i,x_t^i\right) \tag{8}$$

然后根据式 (7) 可以直接求出各特征的模型的期望分布。

但是当使用原生的 sum-product 信念传播算法的时候, 会出现数值溢出的问题。这是因为条件随机场拥有非常大的参数量, 但是这些参数中许多参数对应的权重系数却很小, 这样就导致了模型推断中不断进行 sum-product 操作, 会因为数值过小溢出。为了解决这个问题, 将原来的数值空间转换到 log 空间, sum 就变成了相应的 log-sumexp, product 就变成了求和。而且 logsumexp 不能直接简单地对各值先取 exp 再 sum 最后再取 log, 因为对于很小或者很大的数值, 直接计算会溢出。相应的解决方法为

$$\log\sum_{i=1}^{N}\exp(x_i)=a+\log\sum_{i=1}^{N}\exp(x_i-a) \tag{9}$$

这对任意 a 都成立, 这意味着可以自由地调节指数函数的指数部分, 一个典型的做法是取 x_i 的最大值:

$$a=\max_i(x_i) \tag{10}$$

这样就保证指数最大不会超过 0, 于是就不会上溢。即便剩余的部分下溢了, 也能够得到一个合理的值。

5.2　基于分布式机器学习的系统威胁感知模型

为了提高 RF 算法的性能, 有效解决分布式计算环境下大规模 RF 算法执行过程中的数据通信开销和工作负载不均衡等问题, 本文将改进的随机森林分类算法在 Apache Spark 云计算平台上进一步并行优化, 提出一种基于 Apache Spark 的并行随机森林 (parallel random forest, PRF) 算法。

PR 模型的每棵元决策树都是相互独立构建的, 而且元决策树的每个树节点也是独立划分的。PRF 模型和各个决策树模型的结构使得它们训练过程中的计算任务具有天然的可并行性。

PRF 的双层并行训练过程: 在双层并行训练方法中, 并行训练随机森林模型中各元素决策树模型的构建过程和各元素决策树各节点的分裂过程。由于每个 PRF 模型中的每个元决策树都是通过每个训练子集的独立训练来构建的, 所以每个决策树之间不存在逻辑依赖和数据依赖。因此, 在外部并行训练中, 将训练数据集随机采样到 K 个训练子集中, 分别对这些训练子集进行并行训练, 构建相应的 K 元素决策树模型。在每个元决策树的构建过程中, 通过计算当前特征子集的信息增益率来完成每个节点的分裂过程, 同一层次节点的分裂过程不存在逻辑依赖和数据依赖。因此, 在内层并行训练中, 对每棵决策树中的同一级节点, 分别对当前训练子集的 M 个特征变量同时计算, 以实现节点并行分裂。

在 PRF 模型的每棵元决策树的训练过程中有多种计算任务, 本节根据各计算任务所需的数据资源和数据通信成本, 将这些计算任务分为信息增益率计算任务和节点分裂任务 2 类。

每个决策树模型的训练任务 DAG 包含了对应于决策树模型节点级的多个任务阶段。数据特征降维后, 操作阶段 1 将为 m 个输入特征变量生成 m 个 TGR 任务 (TGR1.1~TGR1.m)。这些

TGR 任务负责计算对应特征变量的信息熵、自分解信息、信息增益和信息增益率，并将计算结果提交给 TNS 1 任务。TNS 1 任务负责寻找最优的拆分特征，并对当前决策树模型的第一个树节点进行拆分。假设y_1是当前阶段的最佳分裂特征，y_1的取值范围为 $\{v_{01}, v_{02}, v_{03}\}$，则第 1 个树节点由特征$y_1$构成，并且生成 3 个子节点，如图 7 所示。拆分树节点后，TNS 1 任务的中间结果被分配到相应的计算节点，以便各计算节点并行计算该决策树的下一级节点分裂。所发送的中间结果包括分裂特征的信息和各个取值$\{v_{01}, v_{02}, v_{03}\}$所对应的数据索引列表。

在作业阶段 2 中，由于y_1是分裂特征，已经在第 1 个节点中被使用，因此接下来根据 TNS 1 的结果为其他特征子集生成新的 TGR 任务。根据$\{v_{01}, v_{02}, v_{03}\}$的数据索引列表，每个特征子集将对应不超过 3 个 TGR 任务。然后将任务的结果提交给任务 TNS 2.1，用于拆分相同的树节点。其他树节点和其他阶段中的任务也以类似方式执行。这样，每个决策树模型训练过程分别建立相应的 DAG 任务调度图，即 PRF 模型的 k 棵决策树，分别建立 k 个 DAG 任务调度图。

本文提出的双层并行训练方法，分别在随机森林模型中的决策树层面和各树的节点层面进行并行化训练。在数据量大的情况下，可以减少模型的训练时间。当数据量增加时，PRF 的性能优势更为明显。

6　高效能数据并行处理与分析系统

融合上述 4 项分布式并行计算关键技术，本文进一步研发了高效能数据处理与智能分析系统，并以天河超级计算机作为主要高性能计算资源池。针对超算调度系统中涉及数据的实际特征较少的困难，研制异构并行环境时空任务调度子系统，解决调度过程中的资源感知差的问题；针对超算调度系统中的资源跨域分配难、策略固定等困难，研制高性能计算资源池及子系统，解决超算平台上的资源自适应低等问题；针对超算平台中缺乏适应超算异构并行的机器学习算法库等缺陷，提出了大数据并行处理与建模子系统，解决了超算算法库中的资源、算力浪费等问题。该系统的研制初步解决了在异构并行超算上构建大数据与人工智能应用环境的问题，有效降低传统企业基于超级计算机、云服务集群等来实现大数据智能分析的使用门槛。高效能数据并行处理与分析系统如图 8 所示。

图 8　高效能数据并行处理与分析系统

Fig. 8　High-performance data parallel processing and analysising system

7　结束语

算力是人工智能应用落地的关键，一直以来是人工智能发展的最大瓶颈。在国家自然科学基金重点项目等课题的资助下，本文从基础理论研究、关键技术突破，到面向领域应用的智能分析系统的研制和应用，形成了面向机器学习的分布式并行计算关键技术体系，研制了高效能数据并行处理与分析系统。该系统及相关研究成果，支撑了中国工程科技知识中心建设项目、广铁集团列车故障快速自动检测与分析系统等多项国家及行业应用项目中大数据和智能算法平台的研制，解决了其算力瓶颈，有力促进了我国人工智能应用技术进步，推动了我国制造、交通、教育、医疗等行业智能软件产品的跨越式发展。项目成果成为了联想、证通电子、东华软件、天闻数媒等上市公司和行业龙头企业行业大数据与智能计算产品的核心组件，解决了其大规模任务调度与资源管理、数据并行处理与智能分析等关键问题。

参考文献：

[1] LU Chienping. Native supercomputing and the revival of Moore's law[J]. APSIPA transactions on signal and information processing, 2017, 6:1–17.

[2] DOHERR D. Supercomputing of tomorrow artificial intelligence in a smarter world[C]// International New York

Conference on Social Sciences. New York, USA, 2017: 1–4.

[3] SHUKUR H, ZEEBAREE S R M, AHMED A J, et al. A state of art survey for concurrent computation and clustering of parallel computing for distributed systems[J]. Journal of applied science and technology trends, 2020, 1(4): 148–154.

[4] CICIRELLI F, GIORDANO A, MASTROIANNI C. Analysis of global and local synchronization in parallel computing[J]. IEEE transactions on parallel and distributed systems, 2020, 32(5): 988–1000.

[5] LU Yuqian, XU Xun, WANG Lihui. Smart manufacturing process and system automation–a critical review of the standards and envisioned scenarios[J]. Journal of manufacturing systems, 2020, 56: 312–325.

[6] LIU Qiang, LENG Jiewu, YAN Douxi, et al. Digital twin-based designing of the configuration, motion, control, and optimization model of a flow-type smart manufacturing system[J]. Journal of manufacturing systems, 2021, 58: 52–64.

[7] KIRIMTAT A, KREJCAR O, KERTESZ A, et al. Future trends and current state of smart city concepts:a survey[J]. IEEE access, 2020, 8: 86448–86467.

[8] LI Kenli Li, LIU Chubo, LI Keqin, et al. A framework of price bidding configurations for resource usage in cloud computing[J]. IEEE transactions on parallel & distributed systems, 2016, 27(8):2168–2181.

[9] ZHONG Jianlong, HE Bingsheng. Medusa: simplified graph processing on GPUs[J]. IEEE transactions on parallel and distributed systems, 2014, 25(6): 1543–1552.

[10] WU Ren, ZHANG Bin, HSU M. GPU-accelerated large scale analytics[EB/OL]. (2009-03-06). http://www.hpl.hp.com/techreports/2009/HPL-2009-38.pdf.

[11] PONCE S P. Towards algorithm transformation for temporal data mining on GPU[D]. Virginia, USA: Virginia Polytechnic Institute and State University, 2009: 805–816.

[12] RAHUL K, BANYAL R K, GOSWAMI P. Analysis and processing aspects of data in big data applications[J]. Journal of discrete mathematical sciences and cryptography, 2020, 23(2): 385–393.

[13] WOLFF J G. The potential of the SP system in machine learning and data analysis for image processing[J]. Big data and cognitive computing, 2021, 5(1): 1–15.

[14] ZHANG Yongpeng, MUELLER F, CUI Xiaohui, et al. GPU-accelerated text mining[C]//Workshop on Exploiting Parallelism Using GPUs and Other Hardware-assisted Methods. Seattle, USA, 2009: 1–6.

[15] SCHATZ M C, TRAPNELL C. Fast exact string matching on the GPU[J]. Center for bioinformatics and computational biology, 2013:1–6.

[16] SCHATZ M C, TRAPNELL C, DELCHER A L, et al. High-throughput sequence alignment using Graphics Processing Units[J]. BMC bioinformatics, 2007, 8(1):1–10.

[17] HE Bingsheng, FANG Wenbin, LUO Qiong, et al. Mars: a MapReduce framework on graphics processors[C]// Proceedings of the 17th International Conference on Parallel Architectures and Compilation Techniques. Toronto, Canada, 2008: 260–269.

[18] MOOLEY A , MURTHY K , SINGH H . DisMaRC: a distributed map reduce framework on CUDA[EB/OL]. http://www.cs.utexas.edu/~karthikm/dismarc.pdf(2019).

[19] KAGERMANN H, WAHLSTER W, HELBIG J. Recommendations for implementing the strategic initiative INDUSTRIE 4.0-Securing the future of German manufacturing industry[J]. Final report of the industrie, 2013, 4: 213−220.

[20] 孙家广. 工业大数据 [J]. 软件和集成电路, 2016 (8):22–23.
SUN Jiaguang. Industrial big data[J].Software and integrated circuit, 2016(8):22–23.

[21] WANG D, LIU Jun, SRINIVASAN R. Data-driven soft sensor approach for quality prediction in a refining process[J]. IEEE transactions on industrial informatics, 2010, 6(1): 11–17.

[22] WAN Jiafu, TANG Shenglong, LI Di, et al. A manufacturing big data solution for active preventive maintenance[J]. IEEE transactions on industrial informatics, 2017, 13(4): 2039–2047.

[23] HONG Sumin, CHOI W, JEONG W K. GPU in-memory processing using spark for iterative computation[C]// 2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing. Madrid, Spain, 2017: 31–41.

[24] JI Feng, MA Xiaosong. Using shared memory to accelerate MapReduce on graphics processing units[C]//International Parallel & Distributed Processing Symposium. Anchorage, USA, 2011: 805–816.

[25] FANG Wenbin, HE Bingsheng, LUO Qiong, et al. Mars: accelerating MapReduce with graphics processors[J]. IEEE transactions on parallel and distributed systems, 2011, 22(4): 608–620.

[26] STUART J A, OWENS J D. Multi-GPU MapReduce on GPU clusters[C]//2011 IEEE International Parallel & Distributed Processing Symposium. Anchorage, USA, 2011: 1068–1079.

[27] ABBASI A, KHUNJUSH F, AZIMI R. A preliminary study of incorporating GPUs in the Hadoop framework [C]//The 16th CSI International Symposium on Computer Architecture and Digital Systems. Shiraz, Iran, 2012: 178–185.

[28] GROSSMAN M, BRETERNITZ M, SARKAR V. HadoopCL: MapReduce on distributed heterogeneous platforms through seamless integration of Hadoop and OpenCL[C]//IEEE International Symposium on Parallel and Distributed Processing, Workshops and PhD Forum. Cambridge, USA, 2013: 1918–1927.

[29] ZHU Jie, LI Juanjuan, HARDESTY E, et al. GPU-in-Hadoop: enabling MapReduce across distributed hetero-

geneous platforms[C]//2014 IEEE/ACIS 13th International Conference on Computer and Information Science (ICIS). Taiyuan, China, 2014: 321–326.

[30] LI Peilong, LUO Yan, ZHANG Ning, et al. HeteroSpark: a heterogeneous CPU/GPU Spark platform for machine learning algorithms[C]//IEEE International Conference on Networking, Architecture and Storage (NAS). Boston, USA, 2015: 347–348.

[31] MANZI D, TOMPKINS D. Exploring GPU acceleration of apache spark[C]//IEEE International Conference on Cloud Engineering (IC2E). Berlin, Germany, 2016: 222–223.

[32] CHOI W, HONG Sumin, JEONG W K. Vispark: GPU-accelerated distributed visual computing using spark[J]. SIAM journal on scientific computing, 2016, 38(5): S700–S719.

[33] ELTEIR M, LIN Heshan, FENG Wuchun, et al. StreamMR: an optimized MapReduce framework for AMD GPUs[C]//17th International Conference on Parallel and Distributed Systems. Tainan, China, 2011: 364–371.

[34] EWEN S, TZOUMAS K, KAUFMANN M, et al. Spinning fast iterative data flows[J]. Proceedings of the VLDB endowment, 2012, 5(11): 1268–1279.

[35] KIM K S, CHOI Y S. Incremental iteration method for fast PageRank computation[C]//Proceedings of the 9th International Conference on Ubiquitous Information Management and Communication. Bali, Indonesia, 2015: 80.

[36] LOW Y, BICKSON D, GONZALEZ J, et al. Distributed GraphLab: a framework for machine learning and data mining in the cloud[J]. Proceedings of the VLDB endowment, 2012, 5(8): 716–727.

[37] XU Yu, KOSTAMAA P. Efficient outer join data skew handling in parallel DBMS[J]. Proceedings of the VLDB endowment, 2009, 2(2): 1390–1396.

[38] TAN Jian, MENG Shicong, MENG Xiaoqiao, et al. Improving reducetask data locality for sequential MapReduce jobs[C]//Proceedings IEEE INFOCOM. Turin, Italy, 2013: 1627–1635.

[39] KWON Y C, BALAZINSKA M, HOWE B, et al. SkewTune: mitigating skew in mapreduce applications[C]//Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data. Scottsdale, USA, 2012: 25–36.

[40] CHEN Qi, LIU Cheng, XIAO Zhen. Improving MapReduce performance using smart speculative execution strategy[J]. IEEE transactions on computers, 2014, 63(4): 954–967.

[41] CHEN Qi, YAO Jinyu, XIAO Zhen. LIBRA: lightweight data skew mitigation in MapReduce[J]. IEEE transactions on parallel and distributed systems, 2015, 26(9): 2520–2533.

[42] LE Yanfang, LIU Jiangchuan, ERGÜN F, et al. Online load balancing for MapReduce with skewed data input[C]//IEEE Conference on Computer Communications. Toronto, Canada, 2014: 2004–2012.

[43] RAMAKRISHNAN S R, SWART G, URMANOV A. Balancing reducer skew in MapReduce workloads using progressive sampling[C]//Proceedings of the Third ACM Symposium on Cloud Computing. San Jose, USA, 2012: 16: 621−633.

[44] GIRSHICK R. Fast R-CNN[C]//2015 IEEE International Conference on Computer Vision. Santiago, Chile, 2015: 1440–1448.

[45] NARODYTSKA N, KASIVISWANATHAN S. Simple black-box adversarial attacks on deep neural networks[C]//2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops. Honolulu, USA, 2017: 1310–1318.

[46] PAPERNOT N, MCDANIEL P, WU Xi, et al. Distillation as a defense to adversarial perturbations against deep neural networks[C]//2016 IEEE Symposium on Security and Privacy. San Jose, USA, 2016: 582–597.

作者简介：

曹嵘晖，副研究员，博士后，主要研究方向为分布式计算与云计算、并行处理体系结构。OpenStack 云计算开源社区核心成员，高性能计算应用软件技术教育部工程研究中心核心成员，湖南省高性能数据处理与智能分析创新团队核心成员。获吴文俊人工智能科技进步一等奖（排名第五）。主持国家重点研发子课题项目 2 项、国家自然科学基金项目 1 项、湖南省自然科学基金项目 1 项，参与撰写湖南省信创云标准 1 项，参与国家重点研发计划项目 2 项、国家自然科学基金重点项目 1 项、面上项目 2 项、湖南省重点研发计划 1 项。申请专利 16 项、授权 7 项，参与撰写专著 1 部，发表学术论文多篇。

唐卓，教授，博士生导师，主要研究方向为分布式计算与云计算。教育部青年长江学者、国家超级计算长沙中心总工程师，担任多个 SCI 期刊的客座编辑，获国家科技进步二等奖（第三）、吴文俊人工智能科技进步一等奖（第一）、中国产学研合作创新成果一等奖（第一）、湖南省技术发明一等奖（第二）。主持科技部国家重点研发计划项目 1 项、国家自然科学基金重点项目 1 项、国家自然科学基金面上项目 2 项、国家自然科学基金应急项目 3 项、国家自然科学基金青年基金项目 1 项，广东省经信委项目、产学研合作项目、中国博士后科学基金等 10 余项。发表学术论文百余篇。

左知微，博士研究生，主要研究方向为分布式机器学习。