doi:10.11887/j.cn.202205011 http://journal.nudt.edu.cn

分布式异构集群中节点优先级调优算法*

胡亚红,邱圆圆,毛家发
(浙江工业大学 计算机科学与技术学院(软件学院),浙江 杭州 310023)

摘 要:节点优先级常用于评价异构集群中节点的性能,因此节点优先级评价指标权重的选择非常重要。采用层次分析法(analytic hierarchy process, AHP)建立了节点优先级评价指标体系,计算得到各指标的初始权重,并使用BP神经网络对初始权重进行优化。训练时,BP网络输入为集群运行中采集的节点实时资源数据,输出为节点的优先级。分析网络训练完成后得到的权重矩阵可以获得各优先级评价指标的优化权重。实验表明,基于AHP和BP的节点优先级评价模型可以更加准确地分析节点性能。相比于Spark默认算法和权重未优化的对照算法,使用调优后的节点优先级可以有效提高集群性能。运行不同工作量的相同负载时,集群平均性能分别提高了16.64%和9.76%;处理相同工作量的不同负载时,集群的平均性能分别提高了12.49%和6.54%。

关键词:层次分析法;BP神经网络;节点优先级;权重;Spark
中图分类号:TP393 **文献标志码**:A **文章编号**:1001-2486(2022)05-102-12

Node priority optimization in distributed heterogeneous clusters

HU Yahong, QIU Yuanyuan, MAO Jiafa
(College of Computer Science and Technology(College of Software), Zhejiang University of Technology, Hangzhou 310023, China)

Abstract: Node priority is often used to evaluate the performance of heterogeneous cluster nodes, and it is of great importance to provide suitable weight for each priority evaluation index. The AHP (analytic hierarchy process) was chosen to establish the evaluation index system of node priority, and the initial weight of each index was calculated. The BP (back propagation) neural network was then used to optimize the weights obtained by using AHP. The input of the BP neural network was the node's performance index values collected during execution of cluster, and the output was the corresponding priority of the node. After the network training, the weight matrix was obtained and used to calculate the optimized weights. The experimental results show that the cluster node priority evaluation model based on AHP and BP can evaluate the node performance more accurately. Compared with the default resource allocation algorithm of Spark and the comparison algorithm with unoptimized weights, the cluster performance is improved effectively by using the node priority optimized. When running the same kind of load with different amount of data, the average cluster performance increases by 16.64% and 9.76%, respectively; and when running different loads with the same amount of data, the average performance of the cluster increases by 12.49% and 6.54%, respectively.

Keywords: analytic hierarchy process; BP neural network; node priority; weight; Spark

Spark是专为大数据处理而设计的快速通用的计算引擎,但是它原生的资源调度策略建立在集群同构的基础之上[1-3]。随着硬件的更迭及高性能部件的引入,集群节点的性能逐渐变得各不相同,集群的异构性日渐凸显[4-5]。Liang等[6]研究发现,同构环境下的资源分配策略不能考虑负载的特性,从而导致原生的Spark调度策略在异构环境中不能充分发挥集群的性能。

目前学者们针对如何提升Spark和Hadoop异构集群性能进行了许多研究[7-9]。郑晓薇等[10]指出Hadoop集群采用原生的任务调度方法,负载不均衡问题常有出现,因此提出了基于节点能力的Hadoop集群自适应任务调度方法,即根据节点的历史和当前负载状态,以节点性能、任务特征等作为节点任务量调度分配的依据,自动地调整任务量。Gunasekaran等[11]考虑Hadoop异构集群特性,提出使用时间约束启动模型策略以减少任务的等待和启动时间。Xu等[12]研究了异构的Hadoop集群下集群节点的动态变化和负载情况,提出采用动态调整的调度算法以实现集群性能优化。虽然都是大数据处理框架,但是Spark和Hadoop在性能研究方面存在着较大差异,

* 收稿日期:2020-11-22
基金项目:国家重点研发计划资助项目(2018YFB0204003)
作者简介:胡亚红(1971—),女,陕西西安人,副教授,博士,硕士生导师,E-mail: huyahong@zjut.edu.cn

Hadoop 集群中所采用方法不完全适用于 Spark 集群。因此,对于提升异构 Spark 集群性能的研究非常重要[13-14]。

樊森[15]考虑异构 Spark 集群下的 Task 调度问题,提出以最小化最大完工时间为优化目标,针对具有 DAG 偏序关系的 Job 和 Stage 以及同一 Stage 中并行 Task 之间的问题特点和特征建立数学模型,优化了 Task 调度算法。基于 Spark on Yarn 模式,Wang 等[16]提出了考虑作业截止日期和值密度的硬实时调度算法以提升集群性能。Kuzmanovska 等[17]则基于多个数据处理框架,提出用 Mesos 反馈控制器动态调整框架权重以进行任务调度。杨志伟等[18]指出 Spark 集群默认的任务调度在异构环境中未考虑节点能力差异,提出了异构 Spark 集群下自适应任务调度策略。该算法监测节点的负载及资源利用率,利用监测得到的参数动态地调整节点的任务分配权重。徐家俊等[19]指出现有的调度策略对于异构 Spark 集群效果不佳,提出了综合考虑任务复杂度、节点性能及节点资源使用情况等因素的分层调度策略。基于集群异构导致计算资源不均衡、Spark 现有的任务调度未考虑集群的异构性以及节点资源的利用情况,胡亚红等[20]提出了基于节点优先级的 Spark 动态自适应调度算法。

神经网络在特征提取和建模方面具有独特优势[21-24]。Zhang 等[25]将 BP 神经网络(back propagation neural network)应用到云计算环境中以进行实时调度,实验证明了算法的有效性。文献[26-27]分别在 Spark 环境中使用神经网络完成了电压偏差预测和交通网络流量预测,均取得了很好的效果。越来越多的研究将 Spark 和深度学习连接在一起,譬如 Moritz 等针对现有流行的批处理计算框架不支持分布式深度学习系统和密集型工作负载的问题,提出了专用于训练深度网络的 Spark 框架 SparkNet[28]。SparkNet 能够大幅提高深度网络的训练效率。

现有研究工作在进行集群资源分配和任务调度时多是基于节点的性能优先级进行。为了得到合理有效的节点优先级,通常需要建立集群节点优先级评价指标体系,并确定各评价指标的权重。指标的权重对于计算节点优先级起着至关重要的作用。理想的指标权重能够赋予节点适合的优先级,有助于进行有效的集群资源调度,从而提升集群性能,减少用户作业执行时间。为了更直观地展示评价指标的权重对于集群性能的影响,本文采用文献[20]中的算法,进行了一组对比实验。实验仅调整了节点性能指标体系中静态因素和动态因素的权重,其他实验配置完全一致。实验结果如表 1 所示。

表 1　节点评价指标权重变化对任务完成时间的影响

Tab. 1　Influence of node evaluation index weights on the completion time of workloads

任务负载类型	任务数据量/GB	任务完成时间/s		时间差距/%
		动静态因素权重比为 0.5∶0.5	动静态因素权重比为 0.3∶0.7	
WordCount	2	63.94	53.10	20.41
WordCount	4	116.33	94.74	22.79
Sort	2	109.19	84.87	28.66

从表 1 可以看出,当节点评价指标权重发生变化时,集群完成相同用户作业的时间差距较为明显。因此非常有必要研究如何更为合理地确定节点优先级评价体系中各指标的权重,从而完成节点优先级的优化。

目前对于优先级评价指标体系中各指标的权重进行优化的研究相对较少。层次分析法[29](analytic hierarchy process,AHP)是一种经典的权重计算方法,但是它的计算结果存在主观性较强的缺点。节点优先级评价指标权重的确定是比较繁杂的非线性工程,需要建立权重的学习机制,可以通过具有学习、记忆、归纳、容错及自适应能力的人工神经网络[30-31](artificial neural network,ANN)予以解决。因此,本文实验平台基于中央处理器(central processing unit, CPU)异构进行验证和说明,文中将层次分析法和 BP 神经网络[32-33]有机地结合在一起,提出了基于 AHP-BP 神经网络模型[34]的集群节点优先级的优化算法。该算法首先采用 AHP 确定节点优先级的评价指标体系,并计算出各指标的初始权重。然后建立 BP 神经网络评价模型并进行网络训练。训练好的 BP 神经网络能够提供更加科学有效的指标权重以优化节点的优先级。

1　构建 Spark 集群节点优先级评价指标体系

AHP 是一种解决多因素多目标问题的决策方法,由美国运筹学家 Saaty 教授于 20 世纪 70 年代首先提出[35-36]。它是将定性与定量相结合的系统分析方法,是一种能将复杂系统的决策思维过程模型化、数量化的方法。AHP 根据系统多目

标要求,把多目标分解成多层次因素,依据因素间关系建立层次结构,进行因素间重要性的两两对比,结合专家打分进行各因素重要性排序从而确定各因素的权重。

1.1 建立递阶层次结构

AHP 的工作流程如图 1 所示。

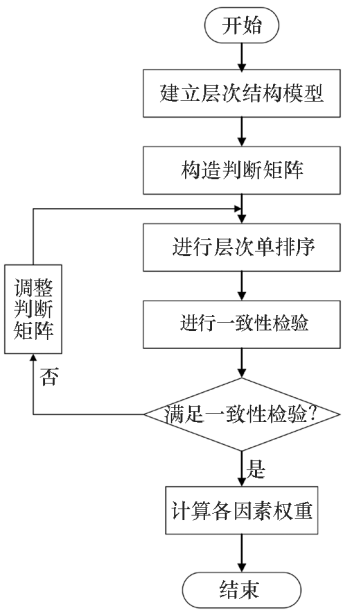


图 1 层次分析法的工作流程

Fig. 1 Workflow of AHP

根据前人文献进行综合分析,本文建立了集群节点优先级评价指标体系,包括 2 个评价因素、8 个评价指标(属性)。

1)静态因素(B_1):

①CPU 速度(C_1):CPU 的时钟频率。计算机的运行速度一般可以由它决定。

②CPU 核数(C_2):一个 CPU 核心的数目。在其他配置相同的条件下,核心数目越多,CPU 的运转速度越快,机器的性能越好。

③内存容量(C_3):节点内存的大小会对 Spark 的运行效率起到决定性作用。

④磁盘容量(C_4):集群运行过程中进行数据的存储及软件的存储。

2)动态因素(B_2):

①CPU 剩余率(C_5):表示节点核心的忙闲情况。CPU 剩余率过低则该节点可用资源很少,需要等资源释放后才可以继续任务的运行。

②内存剩余率(C_6):内存对于集群运行起着极其重要的作用。

③磁盘剩余率(C_7):磁盘用于数据存储。

④CPU 负载(C_8):在一段时间内 CPU 正在处理及等待 CPU 处理的进程数之和,即 CPU 使用队列长度。

节点优先级评价指标体系为三层结构模型:目标层(A)、一级指标层(B)和二级指标层(C),如图 2 所示。

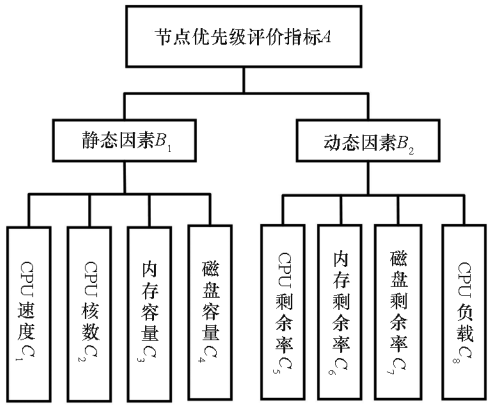


图 2 节点优先级评价指标递阶层次结构

Fig. 2 Hierarchical structure of node priority evaluation

1.2 计算权重

1.2.1 构造判断矩阵

经过专家打分,得到各判断矩阵,如表 2 ~ 4 所示。表 2 给出了一级指标判断矩阵 $A-B$。

表 2 一级指标判断矩阵 $A-B$

Tab. 2 First index judgment matrix $A-B$

A	B_1	B_2
B_1	1	1/7
B_2	7	1

二级指标的判断矩阵分别为 B_1-C(如表 3 所示)和 B_2-C(如表 4 所示)。

表 3 二级指标判断矩阵 B_1-C

Tab. 3 Secondary index judgment matrix B_1-C

B_1	C_1	C_2	C_3	C_4
C_1	1	1/2	1/6	2
C_2	2	1	1/4	2
C_3	6	4	1	8
C_4	1/2	1/2	1/8	1

表 4 二级指标判断矩阵 B_2-C

Tab. 4 Secondary index judgment matrix B_2-C

B_2	C_5	C_6	C_7	C_8
C_5	1	1/3	2	1/2
C_6	3	1	6	1
C_7	1/2	1/6	1	1/4
C_8	2	1	4	1

计算指标单权重,求得二级指标层 C 对一级指标层 B_1 的权重向量为:

$$\boldsymbol{W}_{\mathrm{B1}}^{(2)} = [W_1, W_2, W_3, W_4]^{\mathrm{T}} = [0.113, 0.173, 0.641, 0.073]^{\mathrm{T}} \quad (1)$$

为进行矩阵的一致性检验,计算出一致性指标 $CI = 0.015\ 3$。表 5 给出了不同阶数的矩阵所对应的平均随机一致性指标 RI 的标准值。

表 5　平均随机一致性指标 *RI*

Tab. 5　Average random consistency index *RI*

矩阵阶数	*RI*
1	0
2	0
3	0.58
4	0.90
5	1.12
6	1.24
7	1.32
8	1.41

四阶判断矩阵对应的 RI 为 0.90,则一致性比率 $CR = CI/RI = 0.017 < 0.10$,满足一致性检验。同理可求得另外两个权重向量如下:

$$\boldsymbol{W}_{\mathrm{A}}^{(1)} = [0.3, 0.7]^{\mathrm{T}} \quad (2)$$

$$\boldsymbol{W}_{\mathrm{B2}}^{(2)} = [0.156, 0.422, 0.078, 0.345]^{\mathrm{T}} \quad (3)$$

这两个向量都满足一致性检验。

1.2.2　计算综合权重

使用式(1)~(3),可以得到各评价指标对目标层的综合权重向量为:

$$\boldsymbol{W} = [0.033\ 9, 0.051\ 9, 0.192\ 3, 0.021\ 9, 0.109\ 2, 0.295\ 4, 0.054\ 6, 0.241\ 5]^{\mathrm{T}} \quad (4)$$

此向量也满足一致性检验。

2　建立 Spark 集群节点优先级评价神经网络模型

由于需要通过专家打分确定判断矩阵,AHP 存在一定的主观性和误差。可以利用人工神经网络对 AHP 得到的权重进行优化,从而获得更加客观有效的节点优先级评价体系。

2.1　BP 神经网络

人工神经网络[37-38]是非线性、自适应、自组织的网络系统。本文采用的神经网络模型为 BP 神经网络[39-42],它是一种误差反向传播的多层神经网络,是理论研究比较成熟、应用最为广泛的神经网络之一。图 3 给出了 BP 网络模型。

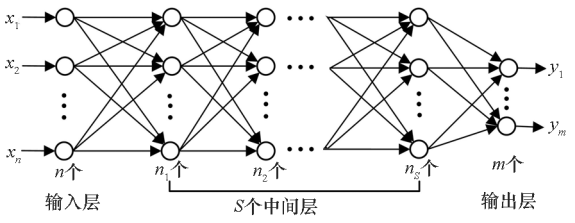


图 3　BP 网络模型

Fig. 3　BP network model

2.2　BP 神经网络的学习算法

BP 网络的学习算法框架如图 4 所示。

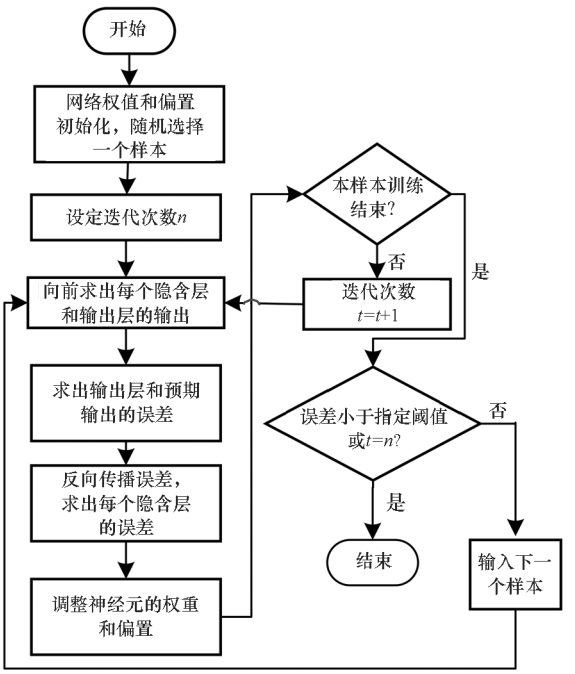


图 4　BP 网络学习算法框架

Fig. 4　Framework of BP network learning algorithm

Step 1:设置初始参数 w_{ij} 和 θ。w_{ij} 表示输入层与隐含层的权系数或隐含层与输出层的权系数,i 表示输入层或隐含层输入节点,j 表示输出节点,θ 表示隐含层各神经元偏置或输出层各神经元偏置。初始设置范围为[0,1]。设定计算精度值 ε 和最大迭代次数 M。

Step 2:将数据样本添加至网络中,利用式(5)计算输出值 y_j:

$$y_j = (1 + \mathrm{e}^{-x})^{-1} \quad (5)$$

其中:$x = \sum_n w_{ij} x_i - \theta$,$x_i$ 表示输入层或隐含层的一个输入节点,$i = 1, 2, \cdots, n$,n 表示输入层神经元数;位于隐含层时,i 用 h 代替,$h = 1, 2, \cdots, S$,S 表示隐含层神经元数。w_{ij} 表示从 i 到 j 的权系数,y_j 为实际计算得到的输出数据。

Step 3:将已知输出数据 d_j 与计算得到的输

出数据进行差值计算,得到两者之间的误差 $\Delta_j = d_j - y_j$,再根据式(6)计算调整权系数的调整量:

$$\Delta w_{ij} = \eta \sigma_j x_i \tag{6}$$

其中,η 为比例系数,即学习率,范围为[0,1],一般设置为 0.01 或者 0.05,具体取值根据实际情况决定。在神经网络中可通过逐步提高 η 的值进行训练,以得到稳定且精度较高的网络。σ_j 表示输出误差相关值。

$$\sigma_j = \eta_j(1 - y_j)(d_j - y_j) = \eta_j \Delta_j (1 - y_j) \tag{7}$$

BP 神经网络的反向误差传播使用式(8)进行:

$$\sigma_j = x_i(1 - x_i)\sum_{k=1}^{m} \sigma_k w_{ik} \quad k = 1,2,\cdots,m \tag{8}$$

其中,m 表示输出层神经元数。

各层神经元的权重调整后结果用式(9)计算得到(其中 t 表示学习次数):

$$w_{ij}(t) = w_{ij}(t-1) + \Delta w_{ij} \tag{9}$$

BP 学习算法不断迭代,对 w_{ij} 值进行调整,直到输出的误差小于允许的误差值 ε 或是达到迭代次数 M,则网络模型训练成功。BP 算法通过迭代算法求得权重。

2.3 AHP-BP 神经网络模型

基于 AHP-BP 神经网络模型的 Spark 集群节点优先级评价算法包括三个部分:数据获取和处理、三层 BP 神经网络模型的建立、神经网络训练和节点优先级评价指标权重的计算。

2.3.1 数据获取和处理

1)节点优先级的初步计算:利用 AHP 计算评价指标体系中各指标的初始权重。

2)节点实时数据的获取:在 Spark 集群中进行任务运行,同时利用 Ganglia[43-45] 实时获取集群节点的资源数据。

3)实时数据的归一化处理:为避免不同优先级评价指标值因量纲差异过大对优先级计算造成影响,需要对采集的节点实时数据进行归一化。本文采用最大 - 最小标准化,即对原始数据进行线性变换,设 min A 和 max A 分别是获取的评价指标 A 的最小值和最大值,将 A 的一个原始值 x 通过最大 - 最小标准化后将映射到区间[0,1]中的值 $\frac{x - \min A}{\max A - \min A}$。各指标归一化后的数据作为神经网络训练样本的输入。

4)初始节点优先级的计算:使用初始权重和归一化后的节点数据计算出节点对应的优先级,此优先级作为神经网络训练样本的输出。

2.3.2 三层 BP 神经网络模型建立

本文建立的 BP 神经网络模型分为三层结构,即输入层、隐含层和输出层。通过不断迭代训练使其收敛可以得到稳定的网络结构和参数。

1)输入层节点的选择:输入节点个数与评价指标的个数相对应。由图 2 可知,评价指标体系共有 8 个评价指标,因此神经网络的输入层节点数 $n = 8$。

2)输出层节点的选择:节点的优先级数值是网络的唯一输出,因此输出层节点数为 1,即 $m = 1$。使用式(4)给出的权重可以计算得到对应的神经网络输出值。

3)隐含层节点的选择:隐含层节点的选择非常关键,因为它会极大影响 BP 神经网络的精确度和学习效率[46]。然而,到目前为止并没有可指导的理论,只能在总结前人文献知识的基础上,根据经验和问题的实际情况而定。通常可以使用式(10)和式(11)来获得隐含层神经元数的大致范围进而确定隐含层的神经元数 S。

$$S = \sqrt{n + m} + \alpha \tag{10}$$

$$\frac{m + n}{2} < S < m + n \tag{11}$$

其中,n 是输入层神经元数,m 是输出层神经元数,α 是[1,10]之间的整数。通过计算,本文建立的神经网络模型中隐含层神经元数 S 的取值范围为[5,8]。

决定系数 R^2 能够评判神经网络模型预测能力的好坏,因此可用来确定隐含层神经元的数目。R^2 的计算方法如式(12)所示。

$$R^2 = \frac{\sum_{i=1}^{N} (\hat{y}_i - \bar{y})^2}{\sum_{i=1}^{N} (y_i - \bar{y})^2} \tag{12}$$

其中,N 表示样本容量,$\hat{y}_i$ 表示模型预测值,y_i 表示样本输出值,$\bar{y}$ 表示样本均值。

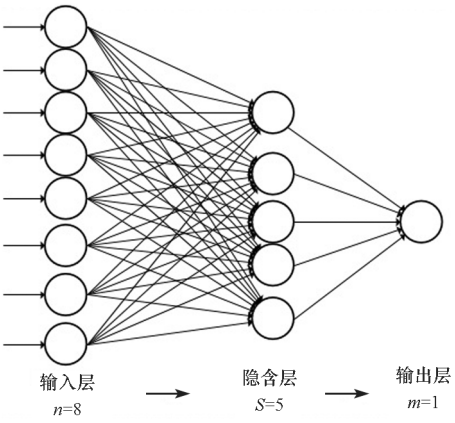
使用式(12)和实际样本的实验测试结果,就可以确定合适的隐含层神经元数。隐含层神经元数取值范围为[5,8],因此设定隐含层神经元数分别为 5、6、7、8,依次进行数据训练得到对应网络的 R^2。实验结果如表 6 所示。

表 6 隐含层神经元数和对应的 R^2

Tab. 6 Number of neurons in the hidden layer and the corresponding R^2

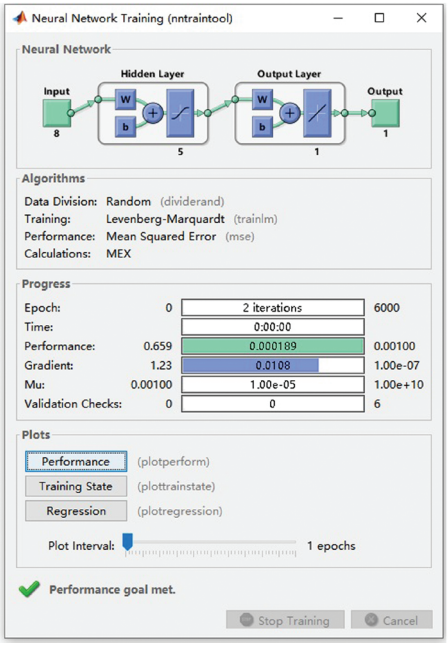
隐含层神经元数	R^2
5	0.89
6	0.87
7	0.74
8	0.70

由表 6 可知,当隐含层神经元数为 5 时,R^2 = 0.89,相比其他神经元的取值效果最好,所以本文模型最终选择 5 作为隐含层神经元数。图 5(a)为节点优先级评价 BP 网络模型,图 5(b) 给出了可视化的网络模型搭建过程。



(a) BP 网络模型

(a) BP network model



(b) 网络模型搭建过程

(b) Building of the BP network

图 5　节点优先级评价 BP 网络模型

Fig. 5　BP network model for node priority evaluation

2.3.3　网络训练和权重计算

网络模型搭建完成后,需要收集学习样本以进行网络训练,样本的选择对于网络的训练很重要。一般需根据实际情况和网络大小来确定合适的样本数,过多或过少都会影响训练结果。依据本文中 BP 网络结构及 Ganglia 得到的节点性能参数进行实验,再根据 R^2 的值即可确定最适合的样本容量。实验结果如表 7 所示。

表 7　样本容量和对应的 R^2

Tab. 7　Sample size and the corresponding R^2

样本容量	R^2
200	0.28
180	0.40
160	0.89
140	0.84
120	0.75
100	0.61

从表 7 可以看出,样本数取 160 时 BP 神经网络具有较好的网络模型预测能力,因此本文中选取 160 个样本进行 BP 网络训练。读取 160 个样本数据进行量化后随机产生训练集和测试集,两者比例为 15∶1,即 150 个训练样本和 10 个测试样本。设置训练参数迭代次数为 6 000,训练要求精度即目标误差为 0.001,学习率为 0.01,最小梯度要求为 e^{-10}。进行网络训练达到收敛状态后获取网络训练权重数据,用矩阵表示如式(13)所示。

式(13)中,权重矩阵的前 8 列分别对应输入层的 8 个神经元,最后 1 列对应输出层神经元;矩阵的各行分别对应隐含层的 5 个神经元。W_{ij}(i = 1,2,…,5; j = 1,2,…,8)表示第 j 个输入神经元与第 i 个隐含层神经元之间的权重;W_{i9}表示输出神经元与第 i 个隐含层神经元之间的权重。

获取的数据需要进行计算以得到最终权重。神经网络训练后得到的神经元权重表示各神经元之间的关系,为了得到输入因素和输出因素之间

$$\boldsymbol{W}=\begin{bmatrix} 0.644\,8 & 0.131\,0 & 0.445\,9 & 0.249\,3 & 0.542\,0 & -0.001\,8 & 0.182\,7 & 0.167\,9 & -0.913\,8 \\ -1.347\,1 & -0.652\,3 & -0.419\,9 & -0.018\,1 & 0.879\,8 & 0.283\,8 & 0.061\,9 & -0.381\,0 & 1.492\,9 \\ -1.226\,3 & 0.696\,0 & 0.930\,9 & -0.137\,1 & -0.059\,1 & 0.879\,4 & 0.309\,6 & -0.280\,3 & 1.840\,9 \\ 0.183\,0 & -0.650\,1 & -0.597\,0 & -0.659\,3 & 0.347\,7 & -0.796\,9 & 0.238\,6 & -1.002\,4 & -0.263\,0 \\ 1.311\,6 & -0.769\,5 & 0.101\,7 & -0.491\,5 & 0.460\,7 & -0.956\,1 & -0.633\,2 & -0.991\,4 & -1.460\,4 \end{bmatrix} \tag{13}$$

的关系,需要对得到的权重系数进行分析处理,利用以下指标[39-40]进行输入因素和输出因素关系的描述。

1)相关显著性系数:

$$r_{ij} = \frac{\sum_{k=1}^{p} W_{ki}(1 - e^{-\omega_{jk}})}{1 + e^{-\omega_{jk}}} \tag{14}$$

2)相关指数:

$$R_{ij} = \left|(1 - e^{-r_{ij}})(1 + e^{-r_{ij}})\right| \tag{15}$$

3)绝对影响系数:

$$S_{ij} = \frac{R_{ij}}{\sum_{i=1}^{n} R_{ij}} \tag{16}$$

其中:i 为神经网络输入层神经元,$i = 1,2,\cdots,n$;j 为神经网络输出层神经元,$j = 1,2,\cdots,m$;k 表示神经网络隐含层神经元,$k = 1,2,\cdots,p$;ω_{jk} 表示输出层神经元 j 和隐含层神经元 k 之间的权重系数。通过上述公式求出绝对影响系数 S_{ij},即为优化的节点优先级评价指标的权重。

使用式(14)~(16),得到本文中各指标的权重,如表 8 所示。

表 8 优化后的节点优先级评价指标权重

Tab. 8 Optimized weights for node priority evaluation index

评价指标	权重
CPU 速度	0.050 4
CPU 核数	0.099 6
内存容量	0.204 3
磁盘容量	0.017 2
CPU 剩余率	0.178 9
内存剩余率	0.318 7
磁盘剩余率	0.091 4
CPU 负载	0.039 5

2.4 SDASA 算法简介

实验采用文献[20]中提出的基于节点优先级的 Spark 动态自适应调度算法(Spark dynamic adaptive scheduling algorithm, SDASA)进行任务调度,其中集群中各个节点的优先级使用 BP 神经网络优化后的节点优先级评价指标权重计算。为方便读者,简要对 SDASA 算法进行介绍。

SDASA 算法[20]的架构如图 6 所示,图中 RPC 为远程过程调用协议,是 Spark 使用的通信框架。

具体算法描述如下:

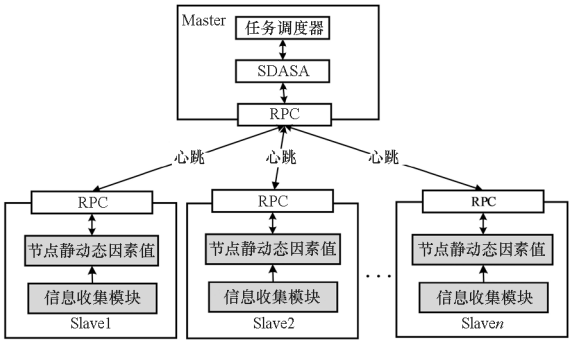


图 6 SDASA 算法架构[20]

Fig. 6 SDASA algorithm architecture diagram[20]

输入:任务集 TaskSet,任务个数为 m;集群节点集合 WorkerOffer,节点个数为 n。

输出:返回任务列表,即第 i 个任务分配到第 j 个节点。

Step 1:启动集群,触发监控启动心跳。

Step 2:利用 Ganglia 获取 Slave 节点动静态因素信息。

Step 3:Master 对集群每个节点进行轮询,将得到的节点的动静态因素数据存入轮询数据库中。

Step 4:Master 对收到的信息进行处理,计算出各节点的动静态指标值。

Step 5:Master 根据每个节点的动静态指标值计算得到各节点优先级。

Step 6:根据各节点优先级对节点集合 WorkerOffer 进行排序,节点优先级越高,排序越靠前;反之,排序越靠后。

Step 7:从优先级最高的节点开始,依次遍历每个节点。在每个节点轮流遍历 TaskSet 中的每个 Task,循环执行 Step 8。

Step 8:获取 Task 在节点上的本地化参数并进行判断,如果参数是最大,执行 Step 9。

Step 9:分配该任务 Task 给该节点。

3 实验结果与分析

为了验证本文提出的节点优先级优化算法的有效性,进行了相应的实验。

3.1 实验环境

基于 Spark on standalone 模式,搭建两个集群进行实验。集群 1 中有 3 个节点,分别为主节点 Master、从节点 Slave1 和 Slave2。集群 2 中有 5 个节点,分别为主节点 Master、从节点 Slave1、Slave2、Slave3、Slave4。集群 1 和集群 2 的节点静态因素指标值分别如表 9 和表 10 所示。

表 9　集群 1 节点静态因素指标值

Tab. 9　Values of static factors for nodes in Cluster 1

节点	CPU 速度/GHz	CPU 核数	内存容量/GB	磁盘容量/GB
Master	2.60	2	5	63.14
Slave1	2.60	2	4	32.01
Slave2	2.60	1	3	32.01

表 10　集群 2 节点静态因素指标值

Tab. 10　Values of static factors for nodes in Cluster 2

节点	CPU 速度/GHz	CPU 核数	内存容量/GB	磁盘容量/GB
Master	2.60	2	5	63.14
Slave1	2.60	2	4	32.01
Slave2	2.60	1	3	32.01
Slave3	2.60	2	4	54.12
Slave4	2.60	1	2	32.01

3.2　BP 神经网络优化的性能评价体系之权重

利用 AHP 计算得到的各指标初始权重如式(4)所示。实验中通过 BP 神经网络优化后的节点优先级计算的各影响因素权重分别为:

1)各因素综合权重如表 8 所示。

2)静、动态因素权重为 0.371 5 和 0.628 5。

3)各静态因素 CPU 速度、CPU 核数、内存容量和磁盘容量的权重分别为 0.135 7、0.268 1、0.549 9 和 0.046 3。

4)各动态因素 CPU 剩余率、内存剩余率、磁盘剩余率、CPU 负载的权重分别为 0.284 6、0.507 0、0.145 4 和 0.062 8。

表 11 和表 12 分别给出了集群 2 中各节点的动态因素指标值随着任务运行的变化情况。实验使用了数据量为 4 GB 的工作负载 Sort,表中 CPU 负载显示的是该节点任务队列的长度。优先级计算使用的是 BP 神经网络优化后的权重。由表可以看到,随着任务的执行,各个节点的优先级均发生了变化。

表 11　任务开始时动态数据采集及节点优先级初始状态

Tab. 11　Initial performance data and priority of each node

节点	CPU 剩余率 /%	内存剩余率/%	磁盘剩余率/%	CPU 负载	节点优先级
Master	34.70	26.18	38.85	1.36	0.21
Slave1	97.80	49.17	58.21	0.24	0.27
Slave2	44.60	24.21	54.23	1.22	0.18
Slave3	69.50	33.91	54.94	0.95	0.25
Slave4	10.30	24.06	32.84	1.48	0.09

表 12　系统运行 28 s 后采集的各节点动态数据及节点优先级

Tab. 12　Performance data and priority of each node after the system runs for 28 s

节点	CPU 剩余率/%	内存剩余率/%	磁盘剩余率/%	CPU 负载	节点优先级
Master	4.60	2.31	38.85	4.16	0.15
Slave1	41.00	41.18	58.21	0.24	0.34
Slave2	8.10	9.28	52.30	1.22	0.14
Slave3	58.30	17.59	53.53	0.95	0.27
Slave4	10.50	3.48	32.84	1.58	0.10

3.3　BP 神经网络权重优化调度算法实验结果及分析

本节展示节点权重优化算法的有效性,对比算法为 Spark 默认调度算法[47-49] 和 2.4 节描述的基于 AHP 权重的 SDASA 算法。采用的负载是中国科学院计算技术研究所研发的基于大数据基准测试的开源性程序集 BigDataBench[50-51]。实验选择了三种工作负载,分别为 WordCount、Sort 和 *K*-means。

实验分别在包含 3 个节点的集群 1 和包含 5 个节点的集群 2 上进行。选取两个集群进行实验的目的是考查随着集群中节点数目的增多,节点权重优化算法对于集群性能提升的影响。两个集群上所进行的实验相同,均为相同数据量不同工作负载实验和相同工作负载不同数据量实验。为了实验数据的准确性,每种实验进行 5 次并记录时间,以 5 次的平均值为最终实验结果。

3.3.1　相同工作负载不同数据量的实验

本实验处理的工作负载是 WordCount,实验使用不同数量的数据集,分别为 2 GB、4 GB、6 GB、8 GB 和 10 GB。图 7 给出了集群 1 上的实验结果,图 8 给出了集群 2 上的实验结果。

由图 7 和图 8 可以得到,运行相同工作负载时,使用 BP 神经网络优化权重的 SDASA 与使用 Spark 默认算法和 AHP 初始权重的 SDASA 相比,任务的执行时间大幅缩减。表 13 给出了具体的对比结果。

从表 13 可以看到,负载相同、工作量不同时,集群 1 环境下,与 Spark 默认算法相比性能平均提升 12.72%;与 SDASA 算法相比平均提升 6.57%。集群 2 中,与 Spark 默认算法相比性能平均提升 16.64%;与 SDASA 算法相比平均提升

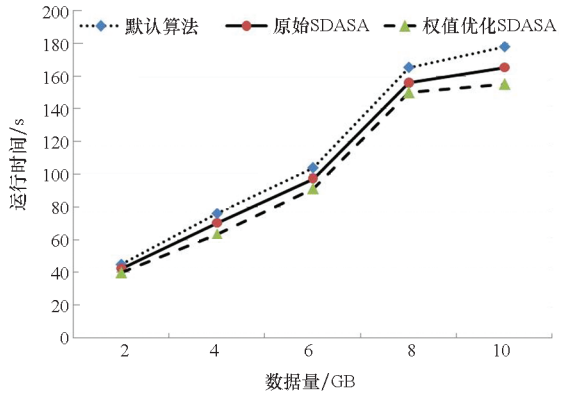


图 7 集群 1 上不同数据量执行时间对比

Fig. 7 Comparison of execution time under different data sizes for Cluster 1

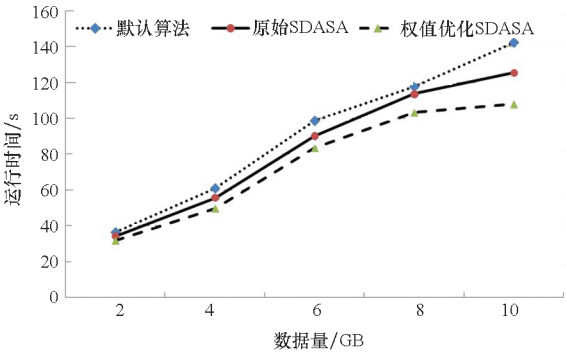


图 8 集群 2 上不同数据量执行时间对比

Fig. 8 Comparison of execution time under different data sizes for Cluster 2

9.76%。随着集群中节点数目的增加,节点优先级优化后,集群的性能提升也更加显著。

表 13 不同数据量时集群性能对比

Tab. 13 Cluster performance comparison under different data sizes

%

集群中节点个数	与 Spark 默认算法比较			与使用 AHP 计算权重的 SDASA 算法比较		
	最大性能提升	最小性能提升	平均性能提升	最大性能提升	最小性能提升	平均性能提升
3	17.12	9.43	12.72	10.08	3.91	6.57
5	24.11	12.19	16.64	13.96	7.46	9.76

3.3.2 不同工作负载相同数据量的实验

为了考察节点性能优先级评价指标的权重对于不同性质任务的影响,对三种不同工作负载进行了实验,所选负载分别为 WordCount、Sort 和 *K*-means,实验使用的数据量均为 2 GB。使用 Spark 默认的任务调度算法、基于 AHP 权重的 SDASA 和本文基于 AHP-BP 权重的 SDASA 得到的实验结果分别如图 9 和图 10 所示。

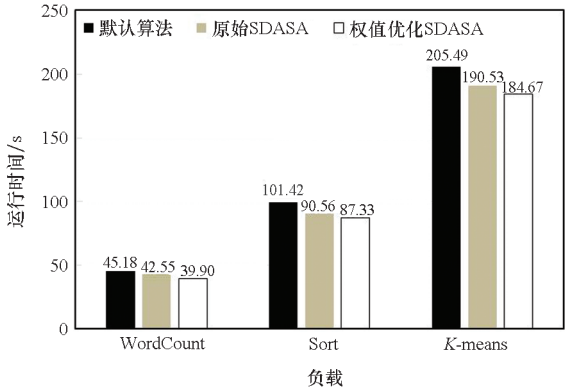


图 9 集群 1 上不同工作负载执行时间对比

Fig. 9 Comparison of execution time under different workloads for Cluster 1

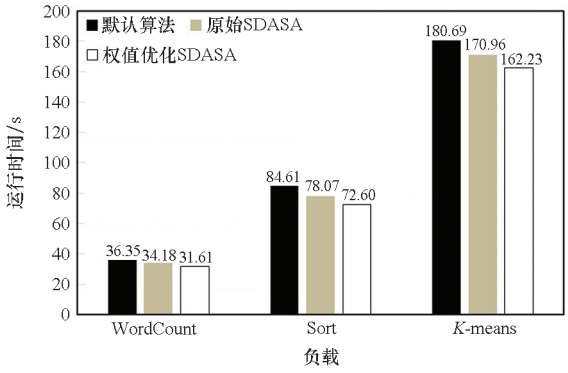


图 10 集群 2 上不同工作负载执行时间对比

Fig. 10 Comparison of execution time under different workloads for Cluster 2

由图 9 和图 10 可以看出,进行节点优先级优化后的 SDASA 与 Spark 默认调度算法和使用 AHP 初始权重的 SDASA 相比,运行 WordCount、Sort 和 *K*-means 三种工作负载时,系统性能都有提升。具体数据见表 14。

表 14 不同工作负载时集群性能对比

Tab. 14 Cluster performance comparison under different workloads

%

集群中节点个数	与 Spark 默认算法比较			与使用 AHP 计算权重的 SDASA 算法比较		
	最大性能提升	最小性能提升	平均性能提升	最大性能提升	最小性能提升	平均性能提升
3	13.89	10.13	11.90	6.24	3.08	4.30
5	14.20	10.22	12.49	7.52	5.10	6.54

从表 14 可以看到,当处理相同工作量的不同负载时,集群 1 环境下,与 Spark 默认算法相比性

能平均提升 11.90%,与 SDASA 算法相比平均提升 4.30%;集群 2 中,与 Spark 默认算法相比性能平均提升 12.49%,与 SDASA 算法相比平均提升 6.54%。同样地,集群的性能提升会随着集群中节点数目的增加而增加。因此可以看出,随着集群规模的进一步加大,本文提出的优先级调优算法能够更好地提高集群性能。

4　结论

为了解决分布式异构集群性能优化问题,本文对集群节点的优先级进行了优化。本文建立了 AHP-BP 神经网络模型,利用 AHP 和 BP 神经网络来确定集群节点优先级评价指标体系中各因素的权重。实验结果表明,使用所提出的节点优先级调优算法能够得到合理的集群节点优先级,提升 Spark 集群系统性能。

下一步的研究工作将从以下几个方面展开:

1)本文模型考虑的异构集群的复杂度主要体现在同一集群的多个节点之间在磁盘容量、CPU 核数和速度以及内存容量等方面存在较大差异,从而导致节点计算能力的不同。本文提出的基于 AHP-BP 神经网络的节点优先级评价模型具有较强的通用性。目前文中的 AHP 层次结构是针对上述的异构模型所建立的。当考虑更为复杂的异构集群,如 CPU-GPU-NPU 等不同类型的异构时,可以将 GPU 及 NPU 的速度、核数、剩余率、负载等因素也作为影响节点性能的因素加以考虑,建立新的 AHP 层次结构模型。同时根据新的节点性能影响因素调整 BP 神经网络的输入,训练符合新 AHP 模型的 BP 网络模型。因此后续将研究本文提出的算法对于其他种类异构集群的有效性。

2)研究更好的学习样本选择方法。

3)选用更合适的神经网络或深度学习方法进行实验研究。

4)搭建大型集群,针对大数据量和多种负载进行更多的实验,以进一步验证优先级优化算法的有效性。

参考文献(References)

[1] ZAHARIA M, XIN R S, WENDELL P, et al. Apache Spark: a unified engine for big data[J]. Communications of the ACM, 2016, 59(11): 56 - 65.

[2] BERNI A. Data-intensive systems: principles and fundamentals using Hadoop and Spark [J]. Computing Reviews, 2020, 61(2): 59 - 59.

[3] ZAHARIA M, CHOWDHURY M, DAS T, et al. Resilient distributed datasets: a fault-tolerant abstraction for in-memory cluster computing [C]//Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation, 2012.

[4] QIN X, JIANG H. A dynamic and reliability-driven scheduling algorithm for parallel real-time jobs executing on heterogeneous clusters[J]. Journal of Parallel and Distributed Computing, 2005, 65(8): 885 - 900.

[5] ZHU X M, HE C, LI K L, et al. Adaptive energy-efficient scheduling for real-time tasks on DVS-enabled heterogeneous clusters[J]. Journal of Parallel and Distributed Computing, 2012, 72(6): 751 - 763.

[6] LIANG Y, TANG Y, ZHU X, et al. Task scheduling strategy for heterogeneous Spark clusters[C]//Artificial Intelligence in China, 2020: 131 - 138.

[7] Apache Hadoop. The Apache software foundation[EB/OL]. (2007 - 09 - 04) [2020 - 09 - 10]. http://hadoop.apache.org/.

[8] KRISH K R, ANWAR A, BUTT A R. hatS: a heterogeneity-aware tiered storage for Hadoop[C]//Proceedings of 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, 2014: 502 - 511.

[9] DAYALAN M. MapReduce: simplified data processing on large clusters [J]. Communications of the ACM, 2008, 51(1): 107 - 113.

[10] 郑晓薇, 项明, 张大为, 等. 基于节点能力的 Hadoop 集群任务自适应调度方法[J]. 计算机研究与发展, 2014, 51(3): 618 - 626.
ZHENG X W, XIANG M, ZHANG D W, et al. An adaptive tasks scheduling method based on the ability of node in Hadoop cluster [J]. Journal of Computer Research and Development, 2014, 51(3): 618 - 626. (in Chinese)

[11] GUNASEKARAN S, SAIRAMESH L, SABENA S, et al. Dynamic scheduling algorithm for reducing start time in Hadoop[C]//Proceedings of the International Conference on Informatics and Analytics, 2016: 1 - 4.

[12] XU X L, CAO L L, WANG X H. Adaptive task scheduling strategy based on dynamic workload adjustment for heterogeneous Hadoop clusters[J]. IEEE Systems Journal, 2016, 10(2): 471 - 482.

[13] ZAHARIA M, CHOWDHURY M, DAS T, et al. Fast and interactive analytics over Hadoop data with Spark[J]. Login, 2012, 37(4): 45 - 51.

[14] SINGH A, KHAMPARIAA, LUHACH A K. Performance comparison of Apache Hadoop and Apache Spark [C]// Proceedings of the Third International Conference on Advanced Informatics for Computing Research, 2019: 1 - 5.

[15] 樊森. 基于异构 Spark 集群下的 Task 调度优化方法[D]. 南京: 东南大学, 2019.
FAN S. Scheduling Spark Tasks to heterogeneous cluster[D]. Nanjing: Southeast University, 2019. (in Chinese)

[16] WANG G L, XU J G, LIU R F, et al. A hard real-time scheduler for Spark on YARN[C]//Proceedings of the 18th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, 2018: 645 - 652.

[17] KUZMANOVSKA A, VAN DEN BOGERT H, MAK R, et al. Achieving performance balance among Spark frameworks with two-level schedulers[C]//Proceedings of 18th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, 2018: 133 – 142.

[18] 杨志伟, 郑烇, 王嵩, 等. 异构 Spark 集群下自适应任务调度策略[J]. 计算机工程, 2016, 42(1): 31 – 35, 40.
YANG Z W, ZHENG Q, WANG S, et al. Adaptive task scheduling strategy for heterogeneous Spark cluster[J]. Computer Engineering, 2016, 42(1): 31 – 35, 40. (in Chinese)

[19] 徐佳俊, 刘功申, 苏波, 等. 异构 Spark 集群的自适应调度策略[J]. 计算机科学与应用, 2016, 6(11): 692 – 704.
XU J J, LIU G S, SU B, et al. Adaptive scheduling strategy for heterogeneous Spark cluster[J]. Computer Science and Application, 2016, 6(11): 692 – 704. (in Chinese)

[20] 胡亚红, 盛夏, 毛家发. 资源不均衡 Spark 环境任务调度优化算法研究[J]. 计算机工程与科学, 2020, 42(2): 203 – 209.
HU Y H, SHENG X, MAO J F. Task scheduling optimization in Spark environment with unbalanced resources[J]. Computer Engineering & Science, 2020, 42(2): 203 – 209. (in Chinese)

[21] 李亚超, 熊德意, 张民. 神经机器翻译综述[J]. 计算机学报, 2018, 41(12): 2734 – 2755.
LI Y C, XIONG D Y, ZHANG M. A survey of neural machine translation[J]. Chinese Journal of Computers, 2018, 41(12): 2734 – 2755. (in Chinese)

[22] ANDOR D, ALBERTI C, WEISS D, et al. Globally normalized transition-based neural networks[C]//Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2016: 2442 – 2453.

[23] 黄佳佳, 李鹏伟, 彭敏, 等. 基于深度学习的主题模型研究[J]. 计算机学报, 2020, 43(5): 827 – 855.
HUANG J J, LI P W, PENG M, et al. Review of deep learning-based topic model[J]. Chinese Journal of Computers, 2020, 43(5): 827 – 855. (in Chinese)

[24] 周飞燕, 金林鹏, 董军. 卷积神经网络研究综述[J]. 计算机学报, 2017, 40(6): 1229 – 1251.
ZHOU F Y, JIN L P, DONG J. Review of convolutional neural network[J]. Chinese Journal of Computers, 2017, 40(6): 1229 – 1251. (in Chinese)

[25] ZHANG G H, BROWN P, LI G B. Research on personal intelligent scheduling algorithm in cloud computing based on BP neural network[J]. Journal of Intelligent & Fuzzy Systems, 2019, 37(3): 3545 – 3554.

[26] ZHANG Y, FANG C, WANG Z F, et al. Voltage deviation forecasting with improved BP neural network[C]//Proceedings of the 2018 International Conference on Mechatronic Systems and Robots, 2018: 37 – 41.

[27] ZHANG Y N, ZHOU Y H, LU H P, et al. Traffic network flow prediction using parallel training for deep convolutional neural networks on Spark cloud[J]. IEEE Transactions on Industrial Informatics, 2020, 16(12): 7369 – 7380.

[28] MORITZ P, NISHIHARA R, STOICA I, et al. SparkNet: training deep networks in Spark[EB/OL]. (2016 – 02 – 28)[2020 – 10 – 10]. https://arxiv.org/abs/1511.06051.

[29] FORMAN E H, GASS S I. The analytic hierarchy process—an exposition[J]. Operations Research, 2001, 49(4): 469 – 486.

[30] 蒋宗礼. 人工神经网络导论[M]. 北京: 高等教育出版社, 2001.
JIANG Z L. Introduction to aritificial neural networks[M]. Beijing: Higher Education Press, 2001. (in Chinese)

[31] 毛健, 赵红东, 姚婧婧. 人工神经网络的发展及应用[J]. 电子设计工程, 2011, 19(24): 62 – 65.
MAO J, ZHAO H D, YAO J J. Application and prospect of artificial neural network[J]. Electronic Design Engineering, 2011, 19(24): 62 – 65. (in Chinese)

[32] RUMELHART D E, HINTON G E, WILLIAMS R J. Learning representations by back-propagating errors[J]. Nature, 1986, 323(9): 533 – 536.

[33] 李友坤. BP 神经网络的研究分析及改进应用[D]. 淮南: 安徽理工大学, 2012.
LI Y K. Research analysis and improved application of BP neural network[D]. Huainan: Anhui University of Science & Technology, 2012. (in Chinese)

[34] 黄仕靖, 陈国华, 吴川徽, 等. 基于改进 AHP-BP 神经网络的科研项目数据库评价指标模型构建[J]. 情报科学, 2020, 38(1): 140 – 146.
HUANG S J, CHEN G H, WU C H, et al. Construction of evaluation index model of scientific research project database based on improved AHP-BP neural network[J]. Information Science, 2020, 38(1): 140 – 146. (in Chinese)

[35] AGRAWAL U, ARORA J, SINGH R, et al. Hybrid wolf-bat algorithm for optimization of connection weights in multi-layer perceptron[J]. ACM Transactions on Multimedia Computing, Communications, and Applications, 2020, 16(1s): 37.

[36] 邓雪, 李家铭, 曾浩健, 等. 层次分析法权重计算方法分析及其应用研究[J]. 数学的实践与认识, 2012, 42(7): 93 – 100.
DENG X, LI J M, ZENG H J, et al. Research on computation methods of AHP wight vector and its applications[J]. Mathematics in Practice and Theory, 2012, 42(7): 93 – 100. (in Chinese)

[37] MICHAEL N. Neural networks and deep learning[EB/OL]. (2019 – 12 – 26)[2020 – 10 – 10]. http://neural networks and deep learning.com/index.html.

[38] 韩永强, 季小慧. 基于 IGA-BP 综合算法的人工神经网络在线损计算中的应用[J]. 中国电力教育, 2011(27): 101 – 103.
HAN Y Q, JI X H. Application of artificial neural network in online loss calculation based on IGA-BP comprehensive algorithm[J]. China Electric Power Education, 2011(27): 101 – 103. (in Chinese)

[39] LIU Q L, SUN P X, FU X Y, et al. Comparative analysis of BP neural network and RBF neural network in seismic performance evaluation of pier columns[J]. Mechanical

Systems and Signal Processing, 2020, 141: 106707.

[40] LI T J, SUN J H, WANG L. An intelligent optimization method of motion management system based on BP neural network[J]. Neural Computing and Applications, 2021, 33(2): 707 - 722.

[41] HU J B, TANG X Q. BP algorithm and its application in artificial neural network[J]. Information Technology, 2004, 28(4): 1 - 4.

[42] 储昭辉, 储文静, 徐立祥, 等. 基于 AHP-BP 神经网络的城市移动图书馆服务质量评价优化模型构建[J]. 图书馆学研究, 2020(10): 19 - 27.
CHU Z H, CHU W J, XU L X, et al. Construction of service quality evaluation optimization model of mobile libraries in urban areas based on AHP-BP neural network[J]. Research on Library Science, 2020(10): 19 - 27. (in Chinese)

[43] SACERDOTI F D, KATZ M J, MASSIE M L, et al. Wide area cluster monitoring with Ganglia[C]//Proceedings of IEEE International Conference on Cluster Computing, 2003: 289 - 298.

[44] DANIEL P, CARLO A, VLADIMIR V, et al. Ganglia monitoring system[EB/OL]. (2018 - 03 - 07)[2020 - 11 - 18]. http://ganglia.info/.

[45] VASWANI A, SHAZEER N, PARMAR N, et al. Attention is all you need[C]//Proceedings of the 31st Conference on Neural Information Processing Systems, 2017: 6000 - 6010.

[46] 沈花玉, 王兆霞, 高成耀, 等. BP 神经网络隐含层单元数的确定[J]. 天津理工大学学报, 2008, 24(5): 13 - 15.
SHEN H Y, WANG Z X, GAO C Y, et al. Determining the number of BP neural network hidden layer units[J]. Journal of Tianjin University of Technology, 2008, 24(5): 13 - 15. (in Chinese)

[47] WANG K W, KHAN M M H, NGUYEN N, et al. Modeling interference for Apache Spark jobs[C]//Proceedings of IEEE 9th International Conference on Cloud Computing, 2016: 423 - 431.

[48] XU Y M, LI K L, HU J T, et al. A genetic algorithm for task scheduling on heterogeneous computing systems using multiple priority queues[J]. Information Sciences, 2014, 270: 255 - 287.

[49] SINGHAL R, PHALAK C, SINGH P. Spark job performance analysis and prediction tool[C]//Proceedings of Companion of the ACM/SPEC International Conference on Performance Engineering, 2018: 49 - 50.

[50] 姜春宇, 孟苗苗. 大数据基准测试流程与测试工具[J]. 信息通信技术, 2014, 8(6): 43 - 46, 51.
JIANG C Y, MENG M M. Test process and tools introduction in big data benchmarking [J]. Information and Communications Technologies, 2014, 8(6): 43 - 46, 51. (in Chinese)

[51] 詹剑锋, 高婉铃, 王磊, 等. BigDataBench: 开源的大数据系统评测基准[J]. 计算机学报, 2016, 39(1): 196 - 211.
ZHAN J F, GAO W L, WANG L, et al. BigDataBench: an open-source big data benchmark suite[J]. Chinese Journal of Computers, 2016, 39(1): 196 - 211. (in Chinese)