基于LSTM和GRNN的容器配额优化算法

周泓岑[1],白 恒[1],才振功[1],蔡 亮[1],顾 静[2],汤志敏[2]
(1. 浙江大学软件学院,浙江宁波 315000; 2. 阿里巴巴集团,浙江杭州 310000)

摘 要: 为了实现容器配额设置自动化和集群资源利用最大化,本文设计了一种容器配额优化算法. 本文在长短期记忆(Long Short-Term Memory,LSTM)神经网络和广义回归神经网络(Generalized Regression Neural Network,GRNN)的基础上设计了深度神经网络(Long short-term memory and GRNN Network,LGN),并使用改进量子粒子群算法优选网络结构超参数,以实现自动调参和更快的收敛速度. 容器配额优化算法步骤如下:首先根据历史数据使用LGN训练资源容量模型,然后使用改进的量子粒子群算法优化模型参数,最后使用资源容量模型计算容器配额. 通过与谷歌容器垂直自动扩展器(Vertical Pod Autoscaler,VPA)和水平自动扩展器(Horizontal Pod Autoscaler,HPA)生成的配额进行对比发现,本文提出的优化算法较VPA和HPA降低了至少10%的资源分配总量,同时提升了至少6%的资源利用率.

关键词: 容器配额; 容量模型; 广义回归神经网络; 长短期记忆神经网络; 量子粒子群算法
中图分类号: TP18;TP393.09 **文献标识码:** A **文章编号:** 0372-2112(2022)02-0366-08
电子学报URL:http://www.ejournal.org.cn **DOI:**10.12263/DZXB.20201211

Container Quota Optimization Algorithm Based on GRNN and LSTM

ZHOU Hong-cen[1], BAI Heng[1], CAI Zhen-gong[1], CAI Liang[1], GU Jing[2], TANG Zhi-min[2]
(1. *College of Software Technology, Zhejiang University, Ningbo, Zhejiang* 315000, *China*;
2. *Alibaba Group, Hangzhou, Zhejiang* 310000, *China*)

Abstract: In order to realize the automation of container quota setting and the maximization of cluster resource utilization, this paper designed a container quota optimization algorithm. In this paper, LGN(long short-term memory and GRNN network) was designed based on the LSTM(long short-term memory) and the GRNN(generalized regression neural network). Also, the improved quantum particle swarm algorithm was used to optimize the hyperparameters of the network structure to achieve automatic parameter adjustment and faster convergence speed. The steps of the container quota optimization algorithm are as follows. First, LGN is used to train the resource capacity model based on historical data. Then the improved quantum particle swarm algorithm is adopted to optimize the model parameters. Finally, the resource capacity model is designed to calculate container quotas. By comparing with the quotas generated by Google Container VPA(vertical pod autoscaler) and HPA(horizontal pod autoscaler), the optimization algorithm proposed in this paper has at least 10% lower total resource allocation and 6% higher resource utilization.

Key words: container quota; capacity model; generalized regression neural network(GRNN); long short-term memory(LSTM); quantum particle swarm optimization

1 引言

随着云计算技术的快速发展,容器作为轻量级的虚拟化技术成为云计算平台虚拟化资源的主要方式,越来越多的公司将应用迁移到容器云平台上,大量数据中心存在资源分配率(即分配给所有容器的资源占可分配总资源的比例)较高而资源利用率较低的情况. 当前业界通常将容器配额设置为标准尺寸或者根据经验进行设置,标准尺寸在大规模系统中并不总是可达,根据经验进行设置需要大量的运维人员,并且这两种方法都存在极大的资源浪费. 不同服务对容器拥有不同的尺寸需求(即容器的各种资源配额),并且服务对资源的需求是弹性的,通常通过预留大量资源以应对服务对资源的弹性需求,所以在满足服务对资源的弹

收稿日期:2020-10-29;修回日期:2021-01-08;责任编辑:王天慧
基金项目:国家重点研发计划(No.2019YFB1600700);阿里巴巴-浙江大学前沿技术联合研发中心(No.XT622019000187);阿里巴巴创新研究计划(AIR)合作项目;浙江大学-中移在线联合创新实验室(No.CMOS01HT20180623)

性需求的同时实现资源节省将是一大难题. 容器尺寸过大容易导致资源碎片和资源利用率不足的问题,容器尺寸过小将导致内存硬盘等资源成为瓶颈并带来额外资源损耗. 因此,容器尺寸的合理设置也是一大难题.

针对这些问题,本文提出了一种新颖的容器配额优化算法,该算法设计了一种新的网络结构用于构建容器配额优化方案. 本文的贡献主要包括以下3个方面.

(1)提出了一种融合长短期记忆神经网络(Long Short-Term Memory,LSTM)和广义回归神经网络(Generalized Regression Neural Network,GRNN)的深度神经网络(LSTM and GRN Network,LGN). 它从时间和数量两个维度捕捉集群状态和资源使用量之间的相互关系,相比其他模型具有更高的准确度.

(2)使用LGN建立了资源容量模型,并基于设计了该模型的容器配额计算方案,与目前常用的谷歌容器垂直自动扩展器(Vertical Pod Autoscaler,VPA)[1]和水平自动扩展器(Horizontal Pod Autoscaler,HPA)相比,该方案减少了至少10%的集群资源分配总量并提升了至少6%的集群资源利用率.

(3)改进的量子粒子群算法(Quantum Particle Swarm Optimization,QPSO)实现了对网络结构超参数的自动化优选和更快的收敛速度,避免了手动调参的烦琐与低效.

2　相关工作

当前云环境下的资源分配问题已经得到学术界广泛的研究[2~5],在云计算平台的资源分配问题上,多数分配策略基于优化算法或者用户及应用分析获得的某些特征. 例如Alam等人提出了一种基于可靠性的云环境资源分配策略,最小化成本的同时,将用户应用可靠性最大化[6];周景才等人则通过分析用户的行为特征来进行资源分配[7];Islam等人提出了一种用于云环境中自适应资源供应的负载预测模型,并将其运用到云环境资源分配领域,且取得了较好的效果[8]. 部分分配策略基于博弈思想,使用博弈模型来解决资源分配问题. 例如丁丁等人提出一种基于双边拍卖机制的适应性云计算资源分配机制[9]. 以上提到的资源分配策略,大多在服务粒度下进行资源分配,容器层面的资源分配策略则主要集中在对容器的实例数和资源请求量进行自动调整. 例如Balla等人提出的Libra自动扩展器,自动为容器设置配额并进行扩缩容管理[10];Rattihalli等人设计了一种基于容器资源利用的自动扩展系统RUBAS,实现了容器的无中断垂直自动扩缩容和资源使用量预估[11];Rossi等人则利用强化学习方法实现了一种云环境的自我管理,根据容器运行情况灵活采用水平扩展或垂直扩展来调整容器的资源分配,然而该方法的模型训练耗时过长,很难运用在生产环境中[12].

作为最大的云服务提供商之一,Google Cloud也为用户提供了容器配额生成服务,称为垂直自动扩展器VPA和水平自动扩展器HPA. VPA可以读取应用容器的资源使用情况等指标,根据目标利用率计算出一个配额推荐值,它会在合适的时机更新所有容器的配额. 但VPA因为稳定性和性能原因尚未被广泛应用. HPA用于在配额恒定的情况下,针对不同的负载对容器的实例数进行伸缩,从而将每个容器的资源使用率控制在一定范围内. HPA相比VPA应用要更加广泛,但不能与VPA同时使用.

3　算法模型LGN

本节将介绍LGN的组成和各部分的结构,它是后文构建资源容量模型的基础. 图1展示了LGN的整体结构,它由广义回归神经网络模块、长短期记忆网络模块、人工神经网络(Artificial Neural Network,ANN)模块和改进的量子粒子群算法模块等部分组成. 容器资源使用量与时间和请求量两个维度都有密切关系. LSTM常用于时间序列预测问题,LGN中LSTM模块通过训练获得下一时刻的资源使用量. GRNN常用于线性与非线性关系拟合,LGN中通过GRNN模块获得请求量与资源使用量之间的相互关系. ANN常用于各种关系的拟合,LGN中通过ANN模块获得最终的资源使用量. 同时LGN使用改进量子粒子群算法对网络结构超参数进行优选.

3.1　广义回归神经网络模块

GRNN的理论基础是非线性回归分析,它可以拟合线性和非线性关系,常用于非线性关系的建模分析[13~15]. 本文设计了一个典型的4层GRNN模型用于拟合服务请求量和资源使用量的相互关系,网络结构图如图2所示. 第一层为输入层,神经元个数为1个,用于输入服务请求量(Queries Per Second,QPS);第二层为模式层,神经元个数为训练样本的个数;第三层为求和层,神经元个数为2个;第四层为输出层,神经元个数为1个,用于输出资源使用量.

3.2　长短期记忆网络模块

为了获得资源使用情况与时间之间的深度关系,本文构建了长短期记忆神经网络,并使用资源使用情况的时序序列对其进行训练. LSTM是一种特殊的循环神经网络,与RNN网络相比,LSTM网络在长时序场景中有更好的表现[16~18]. 本文使用了一个经典的双层LSTM网络模型,输入资源使用量的时间序列,输出下个时间点的资源使用量预测值.

3.3　人工神经网络模块

人工神经网络模块使用BP神经网络作为网络结构[19~22]. 本文设计了一个典型的4层神经网络模型用

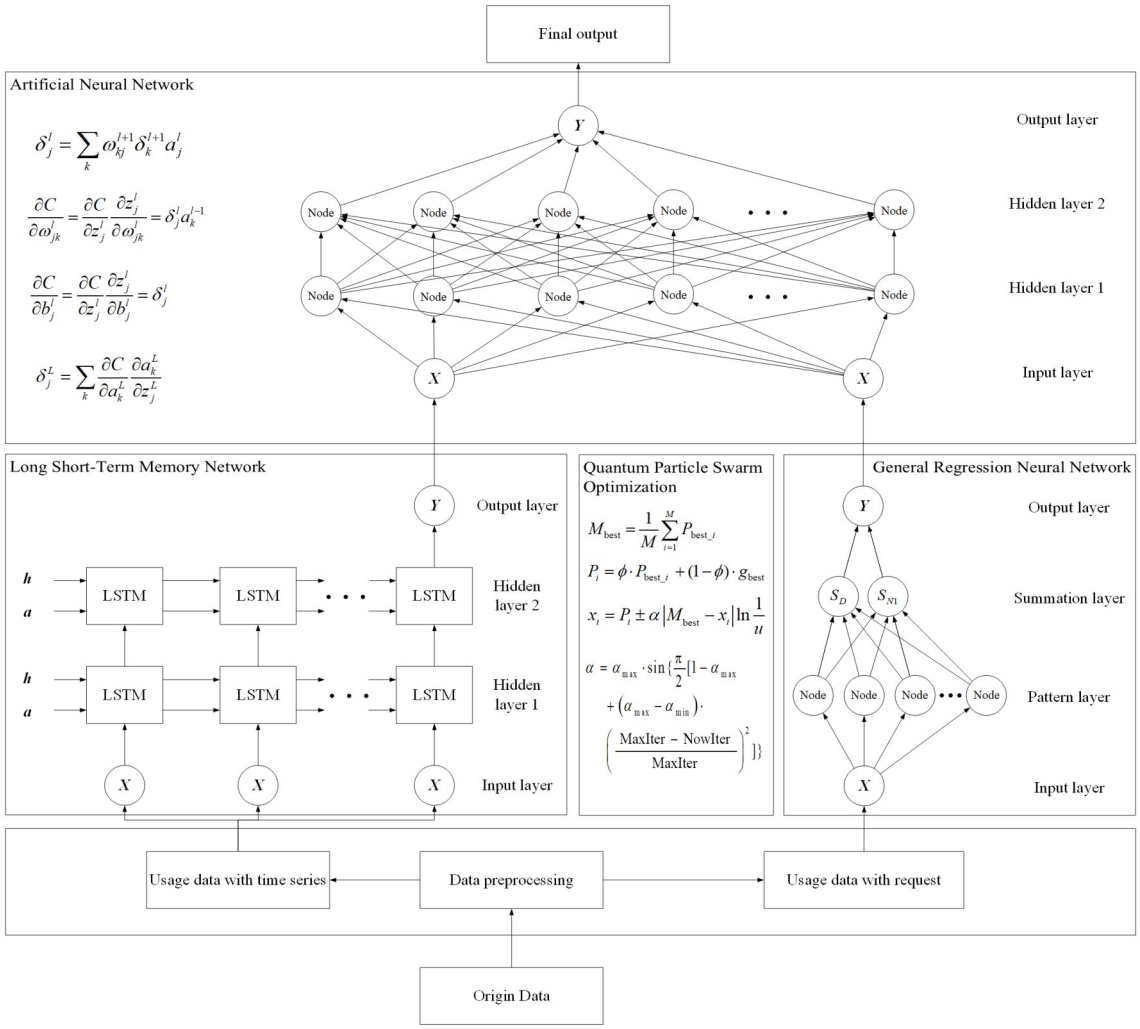

图1 配额优化算法结构图

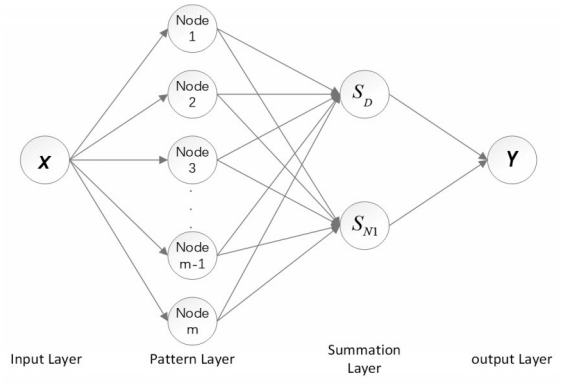

图2 广义回归神经网络结构图

于拟合前2个模块的输出和资源使用量的关系. 神经网络的结构如下:输入层有2个神经元,用于输入GRNN的输出和LSTM的输出,输出层有一个神经元,用于输出资源使用量Y,中间有2层隐藏层,每层隐藏层的神经元个数由改进的量子粒子群算法确定个数.

3.4 改进的量子粒子群算法模块

粒子群算法[23]被广泛应用于各种优化问题中[24,25],相比粒子群优化算法取消了粒子的移动方向属性,该算法可以以概率1收敛于最优解,具有较好的全局寻优能力. 算法的步骤如下.

第1步 计算平均粒子历史最好位置M_{best},计算公式如式(1)所示.

$$M_{\text{best}}=\frac{1}{M}\sum_{i=1}^{M}P_{\text{best\_}i} \tag{1}$$

式(1)中,M表示粒子群的大小,$P_{\text{best_}i}$表示当前迭代中第i个P_{best},P_{best}表示在参数取值的变化过程中到当前为止最优适应度函数值对应的取值.

第2步 粒子位置更新,计算式如式(2)所示.

$$P_i=\phi\cdot P_{\text{best\_}i}+(1-\phi)\cdot g_{\text{best}} \tag{2}$$

式(2)中,g_{best}表示当前全局最优粒子,P_i用于第i个粒子位置的更新. 粒子位置更新式如式(3)所示.

$$x_i=P_i\pm\alpha\left|M_{\text{best}}-x_i\right|\ln\frac{1}{u} \tag{3}$$

式(3)中,x_i表示第i个粒子的位置,ϕ和u为(0,1)上的均匀分布数值. 取+和-的概率都为0.5. α为创新参数,QPSO算法中只有一个创新参数α需要自行设置,α一般取(0,1)之间的固定值,改进后的α计算公式如式(4)所示.

$$\alpha=\alpha_{\max}\cdot\sin\left\{\frac{\pi}{2}\left[1-\alpha_{\max}+\left(\alpha_{\max}-\alpha_{\min}\right)\left(\frac{\text{MaxIter}-\text{NowIter}}{\text{MaxIter}}\right)^2\right]\right\} \tag{4}$$

式(4)中,$\alpha_{\max}$表示最大创新参数,$\alpha_{\min}$表示最小创新参数,NowIter表示当前的迭代次数,MaxIter表示最大迭代次数.

使用改进的量子粒子群算法对GRNN的δ值、LSTM的两层隐含层神经元个数和ANN的两层神经元个数进行优选,设置QPSO算法的各项参数,适应度函数的计算方法如式(5)所示. 使用QPSO算法迭代训练得到LGN的各项网络超参数.

$$\text{Fitness}=\frac{1}{\text{RMSE}} \tag{5}$$

4　配额优化算法

4.1　问题背景

当前各种云计算平台上常见的容器配额配置策略有以下3种.

(1)统一设置为标准尺寸. 在大规模系统上并不总是可达,而且存在显而易见的资源浪费.

(2)交由用户定制. 随机或根据经验进行设定.

(3)按照容器的资源使用量动态规划容器配额. 例如谷歌的Kubernetes VPA,这种纵向扩缩容机制和容器实例数的横向扩缩容相冲突,而且可能会使容器陷入内存溢出死循环.

4.2　问题定义

容器配额优化问题目标是确定分配给每个服务实例(即容器)的资源和总的实例个数,使得所用机器的资源总量最小,实例中资源利用率较高. 配额优化算法问题定义如表1所示.

表1　配额优化问题定义

服务实例数及约束	S:服务集合. P_s:服务s的实例集合P_s. n_s:服务s的实例个数,满足约束条件$n_s^{\min}\leqslant n_s\leqslant n_s^{\max}$.
容器尺寸	$R_c(p)$:实例p的CPU需求,满足$R_c^{\min}(p)\leqslant R_c(p)\leqslant R_c^{\max}(p)$. $R_m(p)$:实例p的内存需求,满足$R_m^{\min}(p)\leqslant R_m(p)\leqslant R_m^{\max}(p)$
机器尺寸	M:机器集合. $C_c(m)$:机器m的CPU容量. $C_m(m)$:机器m的内存容量.
容量模型	Q:服务请求量. U_s^c:服务s的CPU资源使用量. U_s^m:服务s的内存使用量. $F_s^c(Q,T)$:服务请求量Q、时间T和CPU资源使用量映射函数,等于容量模型的训练结果. $F_s^m(Q,T)$:服务请求量Q、时间T和内存资源使用量映射函数,等于容量模型的训练结果.

4.3　算法整体流程

算法整体结构图如图1所示,算法的具体步骤如下.

(1)设定服务的最小容器数$n_s^{\min}$、最大容器数$n_s^{\max}$、最小CPU配额$R_c^{\min}(p)$、最大CPU配额$R_c^{\max}(p)$、最小内存配额$R_m^{\min}(p)$、最大内存配额$R_m^{\max}(p)$. 最小和最大容器数一般根据集群资源使用情况和服务质量要求等进行设定,最小和最大CPU配额和内存配额一般根据服务的资源消耗情况和容器运行最低要求等进行设定.

(2)获取资源使用数据. 获取应用在不同负载梯度下满足服务质量的CPU资源使用总量U_s^c、内存资源使用总量U_s^m、服务请求量Q、时间戳T等数据.

(3)建立容量模型. 使用LGN建立容量模型求得服务请求量Q、时间T与CPU使用量的相互关系$F_s^c(Q,T)$,以及服务请求量Q、时间T与内存使用量的相互关系$F_s^m(Q,T)$.

(4)使用上一步中获得的容量模型,根据服务s、时间T、服务请求量Q计算所需CPU资源总量$D_s^c(Q,T)$和内存资源总量$D_s^m(Q,T)$.

(5)根据(4)中获得的资源需求总量和容器配额算法计算得到容器配额.

4.4　获取历史数据

历史数据包括时间戳、容器的配额、资源使用量、服务的容器数、服务请求量等相关信息.

本文选取了一个具有代表性的开源web应用PeerTube[26]作为测试应用,该应用提供去中心化的视频服

务. 将其容器化后部署到Kubernetes集群中. 采集服务器端应用在不同的每秒请求量(Queries Per Second, QPS)和资源配额下的服务质量和各项运行数据,实验中使用响应时间衡量服务质量.

4.5 建立容量模型

容量模型用于计算在时间和请求量下所需要的资源量,建立容量模型步骤如下.

(1)获取原始数据并进行数据预处理.

(2)建立LGN模型. 建立并使用该模型获得时间T、服务请求量Q与CPU使用量之间的关系$F_s^c(Q,T)$,以及时间T、服务请求量Q与内存使用量之间的关系$F_s^m(Q,T)$.

4.6 容器配额算法

获得服务在某个时间和服务请求量下所需CPU资源和内存资源后,乘以常数N作为所需资源. N的取值通常大于1,如1.15,表示系统为CPU和内存资源提供15%的冗余量,以保证服务运行的稳定性. 接着除以目标利用率得到在该时间和服务请求量下所需资源总量. 之后可以根据以下3种规则求得容器配额.

规则1 最小化容器数. 根据当前时间和服务请求量获得所需资源总量,用所需资源总量除以最小容器数得到每个容器的资源配额. 优点是简单、高效,适用于冷启动. 缺点是容器尺寸较大,影响紧凑部署.

规则2 等比例配额. 使容器配额的CPU和内存比值接近机器的CPU和内存比值. 优点是有利于紧凑部署,适用于冷启动. 缺点是机器结构及容器尺寸分布会影响性能.

规则3 最大化容器数. 根据当前时间和服务请求量获得所需资源总量,用所需资源总量除以最大容器数获得每个容器的资源配额. 优点是简单高效,有利于紧凑部署,适用于冷启动. 缺点是容器太多可能会带来资源管理损耗.

5 方案验证与结果分析

5.1 数据集

本文数据采集主要是基于zabbix系统,采集时间间隔是5 min,每天采集288条数据,包含系统硬件资源使用率数据(CPU、内存、磁盘IO、网络IO等)、QPS、应用性能数据(请求响应时间、错误率)等,本文重点提取了与资源使用量相关的数据采集时间、QPS、CPU使用量和内存使用量,数据格式如表2所示.

表2 数据集格式示意表

时间戳	QPS	内存使用量	CPU使用量
2019/4/1 0:00	52 039	0.892 6	0.78
2019/4/1 0:05	50 301	0.842 4	0.65
2019/4/1 0:10	56 726	0.928 5	0.92

5.2 模型精准度对比

为了验证本文提出的LGN神经网络相较于其他模型有更高的精确度,在同一数据集上使用多种模型进行训练,并比较它们的预测结果.

5.3 模型预测精确度分析

使用不同模型进行预测的精准度对比如表3所示. 结果表明,本文提出的LGN模型相较于其他模型有着更高的预测精确度.

表3 资源使用预测中各种模型的预测误差

模型名	RMSE	MAE	R^2
LGN	121.05	78.67	0.92
LSTM	157.13	92.53	0.65
BiLSTM	176.25	105.24	0.74
GRNN	158.39	102.84	0.79
BP	267.40	155.03	0.54
ARIMA	366.84	167.46	0.82
RF	265.23	145.86	0.89

5.4 实验场景设计

本文所有实验均在云计算平台Kubernetes上进行,采用知名开源视频平台PeerTube作为实验应用.

通过实验采集本文提出的3种配额方案与VPA、HPA在不同负载下的资源使用情况,将应用分为实验组和对照组,实验组包括VPA配额方案、HPA配额方案、等比例配额方案、最大配额数方案和最小配额数方案共5组. 收集实验组应用在不同QPS负载下,在满足服务质量前提下处理请求需要的实例数以及对应的资源使用量. 对照组在实验中维持单实例不限制配额状态,用来为本文提出的算法提供必要参数,并在实验结束后与实验组进行比对. HPA则按照使用惯例及经验,配额设置为1核2 GB,容器资源使用率设置为60%~80%.

最大配额数方案、最小配额数方案和等比例配额方案的实例数原则上由用户设置的参数计算得到,在验证实验中,我们将最大配额数方案实例数设置为10,最小配额数方案实例数设置为3,等比例配额方案的实例数设置为6,VPA配额方案的实例数等同于等比例配额方案,方便实验后进行性能比对.

第1步 部署对照组应用,进行梯度测试,记录测试过程中的QPS与资源使用量,通过前文提出的容量模型确定实验组应用的资源配额.

第2步 使用3种不同配额方案生成3种不同的资源配额,这3种配额方案有最大配额数方案、最小配额数方案和等比例配额方案实例数. 部署开启VPA的应用,实例数等同于等比例实例数. 部署开启HPA的应用,配额恒定,按照容器负载对实例数进行伸缩. 最终得到3种资源配额和VPA的资源配额及HPA资源配额共5种配额方案. 在5台资源容量均等的机器上分别部

署5组实验组应用.

第3步　测试并记录5组实验组应用在处理不同QPS负载时,在满足服务质量的前提下的实例数和资源使用量.

第4步　对比实验组和对照组的各项数据,验证本文算法的有效性.

5.5　实验结果

实验中各个实验组分别部署在总资源为4核8 GB的Kubernetes集群节点上.

使用量子粒子群算法和改进后的量子粒子群算法的迭代次数和适应度值如图3所示. 适应度值越大结果越好,从图中可以看出改进后的量子粒子群算法具有更快的收敛速度,后文的实验均基于改进后的量子粒子群算法.

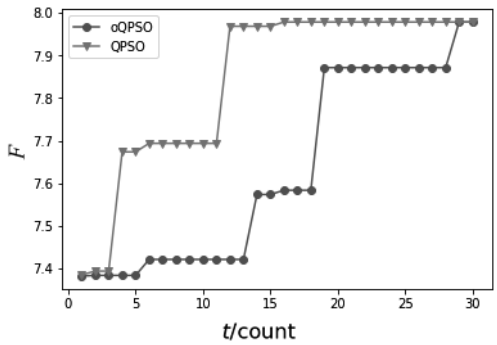

图3　改进量子粒子群算法QPSO和原始量子粒子群算法oQPSO收敛速度对比

等比例配额方案、VPA配额方案、HPA配额方案、最小配额数方案、最大配额数方案在不同负载下的CPU资源配额如图4所示,内存配额如图5所示,CPU资源利用率如图6所示,内存资源利用率如图7所示.

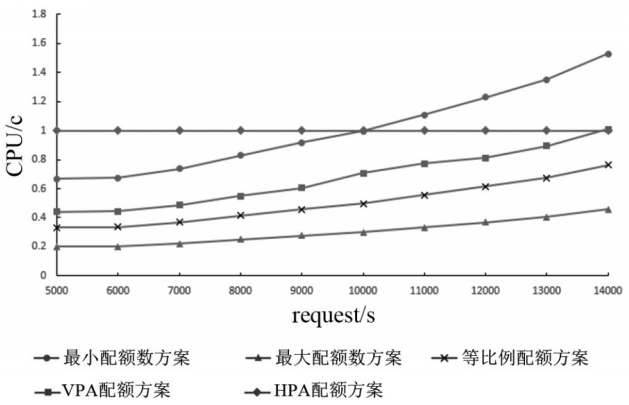

图4　5种配额方案在不同负载下CPU配额

实验中不同配额实验组应用能处理的最大QPS并不相同,在10 000 QPS下五种实验组方案的资源配额,实例数如表4所示,资源分配总量如表5所示,资源利用率如表6所示,HPA实例数变化趋势如图8所示. 实验结果表明,等比例配额方案、最小配额数方案和最大配额数方案

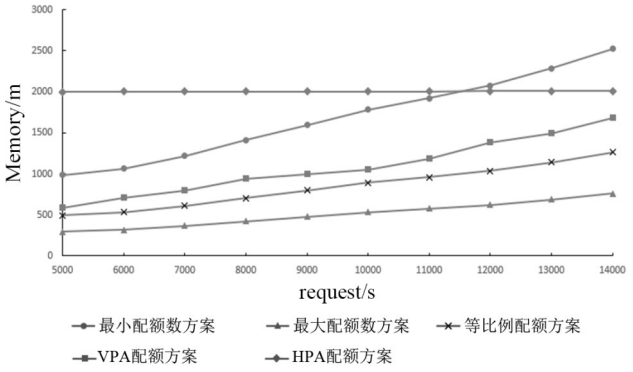

图5　5种配额方案在不同负载下内存配额

同负载下资源分配总量更少,资源利用率更高,且整体资源分配总量相差不大,其中最大配额数方案资源利用率更高,但大量实例数意味着更大的管理和性能开销.

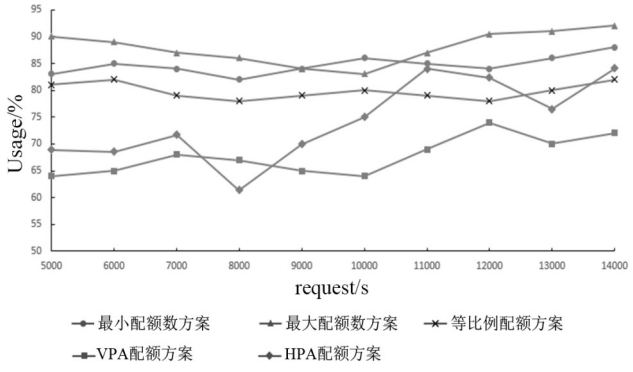

图6　5种配额方案在不同负载下CPU利用率

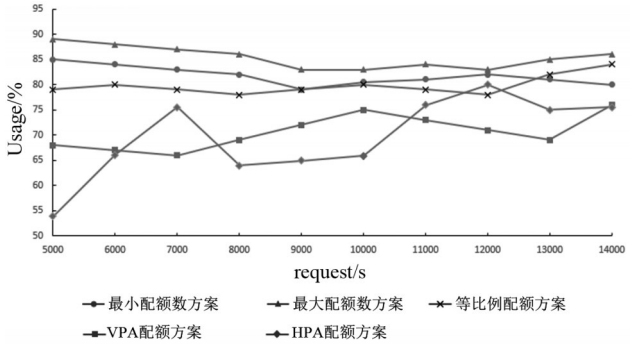

图7　5种配额方案在不同负载下内存利用率

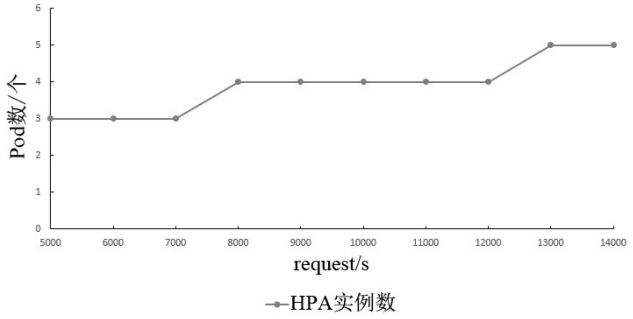

图8　HPA方案实例数随负载变化

表4 同负载下资源配额

实验组	CPU配额	内存配额	实例数
最小配额数方案	0.995 c	1 782 m	3
最大配额数方案	0.298 c	534 m	10
等比例配额方案	0.498 c	891 m	6
VPA配额方案	0.658 c	1 054 m	6
HPA配额方案	1 c	2 000 m	4

表5 同负载下资源分配总量

实验组	CPU总分配	内存总分配
最小配额数方案	2.987 c	5 347 m
最大配额数方案	2.987 c	5 347 m
等比例配额方案	2.987 c	5 347 m
VPA配额方案	3.948 c	6 324 m
HPA配额方案	4 c	8 000 m

表6 同负载下资源利用率

实验组	CPU	内存
最小配额数方案	80.2%	81.6%
最大配额数方案	83.1%	82.7%
等比例配额方案	86.4%	80.5%
VPA配额方案	64.7%	75.2%
HPA配额方案	74.2%	64.3%

5.6 结果分析

从5.5实验结果可以看出,相比于原本的量子粒子群算法,本文改进后的量子粒子群算法具有更快的收敛速度. 在应用配额实验中,随着负载的增加,所有实验组应用的资源配额都在稳步提高,而在不同的负载情况下,不同实验组应用的资源利用率存在一定的波动. 在所有负载情况下,本文提出的3种配额方案在资源利用率上均高于VPA和HPA配额方案.

在处理相同负载的情况下,相比于本文提出的3种配额方案,使用VPA配额方案和HPA配额方案会占用更多的资源,同时资源的利用率也更低,使用LGN生成的3种配额方案占用资源更少,且资源利用率更高,所有应用的服务质量等级均满足要求,这证明了本文提出的配额生成方案相比于VPA配额方案和HPA配额方案,在不影响应用服务质量的前提下,节约了更多资源,有更好的性能表现.

6 总结

本文提出了一种基于广义回归神经网络和长短期记忆神经网络的深度神经网络LGN,用改进量子粒子群算法对网络结构超参数进行优选,使用LGN计算资源容量模型,并设计了基于资源容量模型的容器配额计算方案. 通过基于开源应用系统的实验测试,证明本文提出的方案比Kubernetes自带的VPA和HPA方案提升了超过6%的资源利用率,降低了至少10%的资源分配总量.

使用开源项目验证了所提方法的有效性. 使用本文提出的算法配置容器配额可降低整体资源分配量和提升整体资源使用率. 未来可以使用时序预测算法获得应用的负载变化情况,通过获得未来一段时间的负载情况进行及时的扩缩容,从而实现应用动态扩缩容自动化.

参考文献

[1] CLOUD GOOGLE. Vertical pod autoscaling[EB/OL]. https://cloud. google. com/kubernetes-engine/docs/concepts/verticalpodautoscaler.

[2] JUNG G, SIM K M. Agent-based adaptive resource allocation on the cloud computing environment[C]//2011 40th International Conference on Parallel Processing Workshops. Piscataway, NJ, USA: IEEE, 2011: 345-351.

[3] AKHTER N, OTHMAN M. Energy aware resource allocation of cloud data center: Review and open issues[J]. Cluster Computing, 2016, 19(3): 1163-1182.

[4] YIN L, LUO J, LUO H. Tasks scheduling and resource allocation in fog computing based on containers for smart manufacturing[J]. IEEE Transactions on Industrial Informatics, 2018, 14(10): 4712-4721.

[5] GEETHA P,ROBIN C R R. A comparative-study of load-cloud balancing algorithms in cloud environments[C]// 2017 International Conference on Energy, Communication, Data Analytics and Soft Computing(ICECDS). Piscataway, NJ, USA: IEEE, 2017: 806-810.

[6] ALAM A B M B, ZULKERNINE M, HAQUE A. A reliability-based resource allocation approach for cloud computing[C]//2017 IEEE 7th International Symposium on Cloud and Service Computing(SC2). Piscataway, NJ, USA: IEEE, 2017: 249-252.

[7] 周景才, 张沪寅, 查文亮, 等. 云计算环境下基于用户行为特征的资源分配策略[J]. 计算机研究与发展, 2014, 51(5): 1108-1119.
ZHOU J C, ZHANG H Y, ZHA W L, et al. User-aware resource provision policy for cloud computing[J]. Journal of Computer Research and Development, 2014, 51(5): 1108-1119. (in Chinese)

[8] ISLAM S, KEUNG J, LEE K, et al. Empirical prediction models for adaptive resource provisioning in the cloud[J]. Future Generation Computer Systems, 2012, 28(1): 155-162.

[9] 丁丁, 罗四维, 艾丽华. 基于双向拍卖的适应性云计算资源分配机制[J]. 通信学报, 2012, 33(Z1): 1-143.
DING D, LUO S W , AI L H. Adaptive double auction mechanism for cloud resource allocation[J]. Journal on Communications, 2012, 33(Z1): 1-143. (in Chinese)

[10] BALLA D, SIMON C, MALIOSZ M. Adaptive scaling of Kubernetes pods[C]//NOMS 2020-2020 IEEE/IFIP Net-

work Operations and Management Symposium. Piscataway, NJ, USA: IEEE, 2020: 1-5.
[11] RATTIHALLI G, GOVINDARAJU M, LU H, et al. Exploring potential for non-disruptive vertical auto scaling and resource estimation in kubernetes[C]//2019 IEEE 12th International Conference on Cloud Computing(CLOUD). Piscataway, NJ, USA: IEEE, 2019: 33-40.
[12] ROSSI F, NARDELLI M, CARDELLINI V. Horizontal and vertical scaling of container-based applications using reinforcement learning[C]//2019 IEEE 12th International Conference on Cloud Computing(CLOUD). Piscataway, NJ, USA: IEEE, 2019: 329-338.
[13] AL-MAHASNEH A J, ANAVATTI S G, GARRATT M A, et al. Evolving general regression neural networks using limited incremental evolution for data-driven modeling of non-linear dynamic systems[C]//2018 IEEE Symposium Series on Computational Intelligence(SSCI). Piscataway, NJ, USA: IEEE, 2018: 335-341.
[14] SHAO S, CHEN W , CHEN J. A noisy data regression model based on general regression neural networks[C]// 2011 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2011). Piscataway, NJ, USA: IEEE, 2011: 160-163.
[15] GOULERMAS J Y, LIATSIS P, ZENG X, et al. Density-driven generalized regression neural networks(DD-GRNN) for function approximation[J]. IEEE transactions on neural networks, 2007, 18(6): 1683-1696.
[16] HOCHREITER S, SCHMIDHUBER J. Long short-term memory[J]. Neural computation, 1997, 9(8): 1735-1780.
[17] 谢晓兰, 张征征, 郑强清, 等. 基于 APMSSGA-LSTM 的容器云资源预测[J]. 大数据, 2019, 5(6): 62-72.
XIE X L, ZHANG Z Z, ZHENG Q Q, et al. Container cloud resource prediction based on APMSSGA-LSTM[J]. Big Data Research, 2019, 5(6): 62-72. (in Chinese)
[18] KUMAR S D, SUBHA D P. Prediction of depression from EEG signal using long short term memory(LSTM) [C]//2019 3rd International Conference on Trends in Electronics and Informatics(ICOEI). Piscataway, NJ, USA: IEEE, 2019: 1248-1253.
[19] ASHWINI B C, NIJAGUNARYA Y S. Resource scheduling in cloud computing using back propagation algorithm [C]//2017 2nd International Conference on Emerging Computation and Information Technologies(ICECIT). Piscataway, NJ, USA: IEEE, 2017: 1-6.
[20] KUMAR S, PANDEY M K, NATH A, et al. Missing QoS-values predictions using neural networks for cloud computing environments[C]//2015 International Conference on Computing and Network Communications(CoCoNet). Piscataway, NJ, USA: IEEE, 2015: 414-419.
[21] RAO V, RAO S. Application of artificial neural networks in capacity planning of cloud based IT infrastructure[C]// 2012 IEEE International Conference on Cloud Computing in Emerging Markets(CCEM). Piscataway, NJ, USA: IEEE, 2012: 1-4.
[22] WILSONPRAKASH S, DEEPALAKSHMI P. Artificial neural network based load balancing on software defined networking[C]//2019 IEEE International Conference on Intelligent Techniques in Control, Optimization and Signal Processing(INCOS). Piscataway, NJ, USA: IEEE, 2019: 1-4.
[23] KENNEDY J, EBERHART R. Particle swarm optimization[C]//Proceedings of ICNN'95-International Conference on Neural Networks. Piscataway, NJ, USA: IEEE, 1995: 1942-1948.
[24] 张利彪, 周春光, 马铭, 等. 基于粒子群算法求解多目标优化问题[J]. 计算机研究与发展, 2004, 41(7): 1286-1291.
ZHANG L B, ZHOU C G, MA M, et al. Solving multi-objective optimization problems based on particle swarm optimization[J]. Journal of Computer Research and Development, 2004, 41(7): 1286-1291. (in Chinese)
[25] DING W, FANG W. Target tracking by sequential random draft particle swarm optimization algorithm[C]//2018 IEEE International Smart Cities Conference(ISC2). Piscataway, NJ, USA: IEEE, 2018: 1-7.
[26] PEERTUBE. What is PeerTubef? [EB/OL]. (2020-06-26) [2021-12-30]. https://joinpeertube.org.

作者简介

周泓岑　男,1994年8月出生,四川达州人. 现为浙江大学软件工程硕士研究生. 主要研究方向为云计算、运筹优化、深度学习.

白　恒　男,1995年7月出生,山西吕梁人. 现为浙江大学软件工程硕士研究生. 主要研究方向为云计算、容器化、大数据.
E-mail: zhouhongcen@zju.edu.cn

才振功(通讯作者)　男,1983年出生,河南商丘人. 博士. 现为浙江大学软件学院副研究员,硕士生导师. 主要研究方向为云计算、边缘计算、软件工程.
E-mail: cstcaizg@zju.edu.cn