DOI:10.20033/j.1003-7241.(2026)03-0175-05

云服务器集群资源调度优化方法

刘业辉[1], 陈 炜[2]

(1.信息工程学院 北京工业职业技术学院,北京 100042; 2.北京北控数字科技有限公司,北京 100176)

摘 要:为提升云服务器集群资源利用率,研究基于 docker 容器和 APS 算法的云服务器集群资源调度优化方法。采用 docker swarm 容器架构,通过 Discovery Service 单元监控集群资源节点,获取当前云服务器节点负载的 CPU 和内存利用率;利用线性回归模型预测节点负载,计算每个云服务器集群节点权重;选择权重相同的云服务器节点,使用 APS 算法中的蚁群算法进行优化,并获取云服务集群资源调度优化结果;依据该结果 docker swarm 容器启动 Leadership 单元,为用户创建融合并分配最佳云服务器集群节点,从而实现云服务器集群资源调度优化。仿真实验表明:该方法具备较为准确的云服务器集群节点负载预测能力的同时,还可有效对云服务器集群资源进行调度优化,有效提升了云服务器集群资源利用率,应用效果较为显著。

关键词:docker 容器;APS 算法;蚁群算法;云服务器;集群资源

中图分类号: TP301.6 **文献标志码**: A **文章编号**: 1003-7241(2026)03-0175-05

Optimization method for cloud server cluster resource scheduling

LIU Yehui[1], CHEN Wei[2]

(1. College of Information Engineering, Beijing Polytechnic College, Beijing 100042, China;
2. Beijing BG Digital Technology Co., Ltd., Beijing 100176, China)

Abstract: To improve the resource utilization of cloud server clusters, this paper researches on optimization method for cloud server cluster resource scheduling based on docker container and APS algorithm. It Adopts docker swarm container architecture, monitors cluster resource nodes through Discovery Service units to obtain CPU and memory utilization of current cloud server node load. Using a linear regression model to predict node load and calculate the weight of each cloud server cluster node, it selects cloud server nodes with the same weight, uses the ant colony algorithm in the APS algorithm for optimization, and obtains the optimization results of cloud service cluster resource scheduling. Based on this result, the docker swarm container starts the Leadership unit, creates a fusion and allocates the best cloud server cluster node for users, thereby achieving optimization of cloud server cluster resource scheduling. Simulation experiments show that this method not only has the ability to accurately predict the load of cloud server cluster nodes, but also can effectively optimize the scheduling of cloud server cluster resources, effectively improve the utilization rate of cloud server cluster resources, and the application effect is significant.

Keywords: docker container; APS algorithm; ant colony servers; cloud server; cluster resources

在云计算环境中,资源调度优化不仅要考虑服务器上的资源利用率[1,2],还要考虑用户的需求和服务质量指标。通过综合考虑不同的性能指标和限制条件,如处理能力、内存、网络带宽等,资源调度算法可以更好地满足用户的需求,并提高云计算系统的整体性能。资源调度优化的目标是实现自动化、资源优化、简洁管理和虚拟资源与物理资源的整合。目前学者谢雍生等[3]提出容器集群自均衡调度算法,该算法通过建立 docker 容器集群资源利用率估计模型,通过该模型定义集群节点状态空间、奖励函数等,通过改进的深度强化学习算法(deep q-network, DQN)算法实现集群的调度优化。Kouser 等[4]提出云网络聚类的资源优化调度算法,该方法以当前云计算资源中的主机属性作为基础,通过对主机属性进行聚类后,生成集成矩阵,再对每个类簇内主机负载进行运算后,将负载较大的主机内资源动态调整到负载较小的主机上,实现云服务器集群的资源调度优化。Uma 等[5]提出深度强化 Q 学习优化智能资源调度方法,该方法依据当前云计算集群节点的负载,考虑云计算集群的安全约束,使用深度强化 Q 学习优化算法实现集群内节点负载的优化调度。刘陈伟等[6]提出数据能耗的云计算集群调度优化算法,该算法通过预测云计算集群中每个节点的能耗,通过改进粒子群算法按照节点能耗对云计算集群进行调度优化。以上方法在实际应用过程中均取得了一定的成果,但均存在云服务集群资源调度冲突现象。

收稿日期:2024-07-11;**录用日期**:2024-08-24

基金项目:北京工业职业技术学院信息工程学院课题(BGY2022-KY19Z)

作者简介:刘业辉(1969—),男,硕士,教授,研究方向:通信工程、移动互联应用、人工智能技术应用等。

引用本文:刘业辉, 陈炜. 云服务器集群资源调度优化方法[J]. 自动化技术与应用, 2026,45(3):175-179. (LIU Yehui, CHEN Wei. Optimization method for cloud server cluster resource scheduling[J]. Techniques of Automation and Applications, 2026,45(3):175-179.)

本文以 docker 容器和智能调度算法(aps algorithm, APS)算法为基础,研究基于 docker 容器和 APS 算法的云服务器集群资源调度优化方法研究,以提升云服务器为用户服务能力。

1 云集群资源调度优化仿真

1.1 docker swarm 容器架构

docker swarm 是版本 Docker1.1.2 之前的一个独立项目,目前 docker 容器升级后,其与该容器合并后,形成新的 docker swarm 容器,其是新版本支持 docker 容器云服务器集群的管理工具。当云服务器用户通过应用程序编程接口(application programming interface, API)发送指令时,docker swarm 容器负责管理云服务器内 docker 容器内引擎节点组成的集群,并按照集群配置和约束规则进行调度。在此架构云服务器的 docker swarm 容器,其结构如图 1 所示。

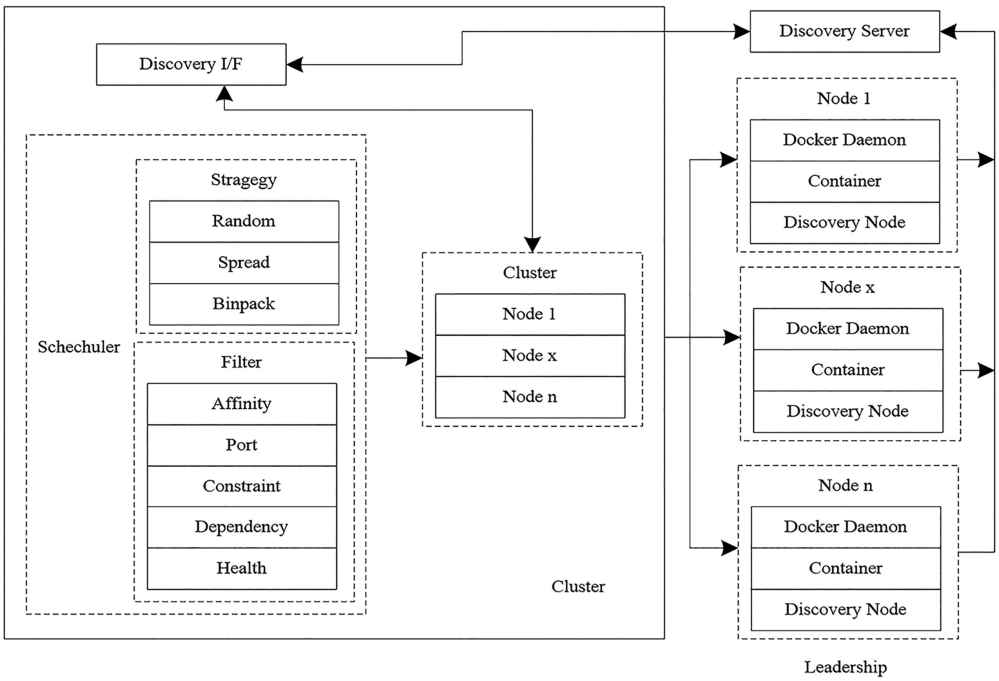

图 1 云服务器 docker swarm 容器架构示意图

Fig. 1 Schematic diagram of cloud server docker swarm container architecture

云服务器 docker swarm 容器架构由 Discovery Service 单元、Scheduler 单元和 Leadership 单元组成,其中 Discovery Service 负责获取云服务器的 docker swarm 容器内的节点,起到在云服务器中的节点监视功能;Scheduler 单元内包含 Stragegy、Filter 等不同类型过滤器,针对云计算集群内节点键值对节点标签进行过滤约束,并可针对用户不同需求执行存储驱动、系统操作等功能;而 Leadership 单元负责为云服务器的用户创建融合并分配最佳节点,是实现云服务器集群资源调度的执行单元。

Leadership 单元调度云服务器集群资源过程如图 2 所示。

云服务器 docker swarm 容器的 Leader-ship 单元接收到集群资源调度请求时,该请求内部会包含若干个 constraint,需要为其分配合理的内存,调度含有 prodution 标签节点,以及为其分配未被占用的 Node,然后将所有的 Node 作为一个 List,传输到一个 filter 链内,通过 filter 链过滤掉不符合条件的 Node,再从剩余的 Node 里,按照调度优化需求[7-9],筛选出最佳 Node,实现云服务器集群资源调度优化。

1.2 APS 算法资源调度优化策略

以云服务器 docker swarm 容器为基础,设计云服务器内集群资源进行调度优化策略,该策略首先动态收集云服务器内节点信息,依据当前集群节点信息,预测下一个时刻云服务器内节点资源使用状况,依据所预测的节点使用状况,对权重相同的节点重新计算权值,然后使用 APS 算法中的蚁群算法选择最优权值,依据最优权值对应的节点启动和运行 docker swarm 容器,实现云服务器集群资源调度优化。

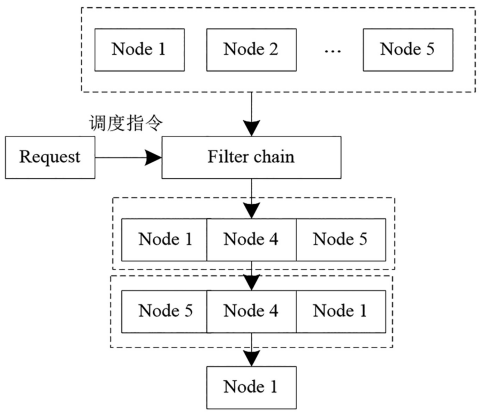

图 2 Leadership 单元调度云服务器集群资源示意图

Fig. 2 Leadership unit scheduling cloud server cluster resource diagram

1.2.1 线性回归节点负载预测

在云服务器 docker swarm 容器内,使用 Discovery Service 单元内的监控组件获取当前云服务器的 CPU 占用率、内存利用率信息后,建立数据集,由 C 和 M 表示,然后将以上两个数据集作为输入,利用一元线性回归模型对云服务器集群节点的负载进行预测,其详细过程如下。

Discovery Service 单元内的监控组件获取前 t_n 个时刻的云服务器集群节点 CPU 占用率表达公式为

$$\boldsymbol{C} = (\langle t_1, c_1\rangle, \langle t_2, c_2\rangle, \cdots, \langle t_n, c_n\rangle) \tag{1}$$

式中,$\boldsymbol{C}$ 为 CPU 占用率向量,t 为时刻变量,n 为时刻序号,t_n 为第 n 个时刻的时间常量,c_n 为第 n 个时刻的 CPU 占用率变量。

获取前 t_n 个时刻的云服务器集群节点内存利用率表达公式为

$$\boldsymbol{M} = (\langle t_1, m_1\rangle, \langle t_2, m_2\rangle, \cdots, \langle t_n, m_n\rangle) \tag{2}$$

式中,$\boldsymbol{M}$ 为内存利用率向量,m 为内存利用率变量。

以式(1)、(2)为基础,使用一元线性回归分析模型对其进行模拟,输出时可为 t_{n+1} 时的云服务器集群节点 CPU 占用率 c_{n+1} 和内存利用率 m_{n+1}。

假设在时刻为 t_i,此时 Discovery Service 单元内的监控组件获取的监控对象值为 r_i,则时间序列 $T = \{t_1, t_2, \cdots, t_n\}$ 与监控对象的取值序列 $R = \{r_1, r_2, \cdots, r_n\}$ 之间存在 r 关于 t 的一次函数,表达公式为

$$r(t) = h_1 t + h_2 \tag{3}$$

式中,r 为监控对象值变量,h_1、h_2 均为不依赖 t 的常数。

令 ε 表示云服务器集群节点负载的真实值和预测值之间的误差,那么每个节点负载真实值和预测值之间的误差均相互独立,且服从同一分布,该误差计算公式为

$$\varepsilon = r - (h_1 t + h_2),\ \varepsilon \sim N(0, \sigma^2) \tag{4}$$

式中,ε 为误差变量,N 为正态分布符号,σ^2 表示方差。

依据公式(4)建立云服务器集群节点负载预测一元线性回归模型,表达公式为

$$r(t) = h_1 t + h_2 + \varepsilon \tag{5}$$

常数 h_1、h_2 对预测云服务器集群节点负载影响较大,需要确立该两个常数数值,由于云服务器集群节点样本 r 和监控值的随机误差 ε 为相互独立关系[10],建立监控云服务器集群节点样本的联合密度函数 H。

$$H = \left(\frac{1}{\sqrt{2\pi}}\right)^n \mathrm{e}^{-\frac{\sum_{i=1}^{n}(r_i - at - b)^2}{2\sigma^2}} \tag{6}$$

式中,H 为联合密度函数,a、b 为待估参数。

设置公式(6)的最大值,则可得到常量 h_1、h_2 的极大似然估计值,表达公式为

$$\begin{cases} \tilde{h}_1 = \dfrac{\sum_{i=1}^{n}(t_i - \bar{t})(r_i - \bar{r})}{\sum_{i=1}^{n}(t_i - \bar{t})^2} \\ \tilde{h}_2 = \bar{r} - \tilde{h}_1 \bar{t} \end{cases} \tag{7}$$

式中,$\tilde{h}_1$、$\tilde{h}_2$ 表示常量 h_1、h_2 的极大似然估计值;$\bar{t}$、$\bar{r}$ 分别表示时间序列和云服务器集群节点样本监控值的均值。

依据公式(7)结果,得到 r 关于 t 的经验回归方程,表达公式为

$$\hat{r} = \tilde{h}_1 t + \tilde{h}_2 \tag{8}$$

其中,$\hat{r}$ 表示 r 关于 t 的经验回归方程。

将时刻 t_{n+1} 代入到上述公式内,可得到云服务器集群节点在该时刻对应的云服务器集群节点 CPU 占用率 c_{n+1} 和内存利用率 m_{n+1} 数值,其表达公式为

$$\hat{r}_{n+1} = \tilde{h}_1 t_{n+1} + \tilde{h}_2 \tag{9}$$

经过上述步骤,利用公式(9)即可得到云服务器集群节点在下一个时刻的 CPU 占用率和内存利用率,得到其在运行过程中的负载参数。接下来需要依据下一个时刻云服务器负载参数对其集群资源进行调度优化。

1.2.2 APS 算法资源调度优化

在 APS 算法中,蚁群优化算法属于一种仿生概率型优化算法,其具备信息正反馈以及启发式搜索特点[11,12],在推荐、优化等方面应用极为广泛,在此以 APS 算法中的蚁群优化算法实现云服务器集群节点负载的寻优,其详细过程如下。

计算第 t_n 时刻云服务器集群节点 r_i 的权值 ω_i,表达公式为

$$\omega_i = \alpha \boldsymbol{C}(r_i) + \beta \boldsymbol{M}(r_i) \tag{10}$$

式中,ω_i 为第 i 个节点的权值,α、β 表示用户自定义云服务器集群节点在 CPU 占用率和内存利用率的权重,且 $\alpha + \beta = 1$。

利用公式(10)计算云服务器集群节点的负载权重后,再以公式(9)结果为基础,重新计算节点权重相同的节点权重,然后使用蚁群寻优算法对节点权重相同云服务器集群节点进行寻优,其寻优过程如下。

在云服务器 docker swarm 容器内,标记权重相同的云服务器集群节点,将每节点看作一只人工蚂蚁,其目的在云服务器集群节点上的任务调度到可用节点上[13-15],实现资源的负载均衡,蚂蚁在该过程中观察每个节点上的资源信息,按照信息素轨迹寻找资源,则第 j 个云服务器集群节点上的资源 $\boldsymbol{R}(j)$ 表达公式为

$$\boldsymbol{R}(j) = \frac{\boldsymbol{\Phi}_{\mathrm{m}} r'_{\mathrm{m},j}}{r_{\mathrm{m},j}} + \frac{\boldsymbol{\Phi}_{\mathrm{p}} r'_{\mathrm{p},j}}{r_{\mathrm{p},j}} \tag{11}$$

式中,$\boldsymbol{R}(j)$ 为第 j 个节点的资源向量,$r'_{\mathrm{m},j}$、$r_{\mathrm{m},j}$ 分别表示云服务器集群节点 j 的可用内存、总内存;$r'_{\mathrm{p},j}$、$r_{\mathrm{p},j}$ 分别表示云服务器集群节点 j 的可用 CPU 和总 CPU;$\boldsymbol{\Phi}_{\mathrm{m}}$、$\boldsymbol{\Phi}_{\mathrm{p}}$ 分别表示当前 docker swarm 容器内存和 CPU 大小。

将公式(11)结果看作蚂蚁的初始信息素浓度,令 $p(t,j)$ 表示云服务器集群节点 j 被选择的概率,依据该概率则下一个云服务器集群节点 j 表达公式为

$$j = \begin{cases} \arg\max p(t,j), & q < q_0 \\ p(t,j), & \text{其他} \end{cases} \tag{12}$$

其中，q_0 表示蚂蚁探索率阈值，q 为当前蚂蚁探索率。

计算蚂蚁在寻优过程中[16-20]，每个云服务器集群节点信息素轨迹消失值，表达公式为

$$\tau(g)=\begin{cases}[\boldsymbol{R}(j)+\Delta]-\zeta[\boldsymbol{R}(j)+\Delta], j\in Q(k)\\ \tau(t-1)-\zeta\tau(t-1), \text{其他}\end{cases} \tag{13}$$

式中，$\tau(g)$ 表示步数为 g 时，云服务器集群节点信息素轨迹消失值；$Q(k)$ 表示第 k 个 docker swarm 容器内需要调度需要的资源；Δ 表示云服务器集群节点信息素变化量；ζ 为当前信息素浓度。

令 $\boldsymbol{Q}_w$ 表示云服务器集群节点最佳优化调度结果，其表达公式为

$$\boldsymbol{Q}_w=\arg\max\sum_{k=1}^{n}[Q(k)=x]\cdot\tau(g) \tag{14}$$

式中，$\boldsymbol{Q}_w$ 为最佳优化调度结果向量，n 为需要调度优化资源总数，$k\in n$，x 为蚂蚁选择的最佳云服务器集群节点。

利用公式(14)即可在权值相同的 docker swarm 容器内云服务器集群节点中获取最优节点，通过运行 docker swarm 容器 Leadership 单元驱动该最优节点，实现云服务器集群资源调度优化。

2 仿真实验

为验证本文方法的有效性，搭建仿真实验环境，docker swarm 环境运用 Go 编译器编写，实验平台架构为 X86-64，云提供商选择华为云，在华为云上生成 SwarmKit 集群，该集群管理节点为 2 个，每个管理具备 10 个工作节点，工作节点编码由 1-10 和 11-20 分属 2 个集群管理节点。该两个管理节点与其下属工作节点的资源配置相同，其资源配置情况如表 1 所示。

在仿真实验过程中，管理节点的可用性数值为 0，运用每个工作节点分别连接一台用户主机，每台主机随机发起任务，运用本文方法对该云服务器集群资源进行调度优化仿真，验证本文方法应用效果。

表 1 云服务器集群节点资源配置

Tab. 1 Cloud server cluster node resource configuration

节点编码	内存/G	CPU
1	1	1 核
2	1	2 核
3	0.5	1 核
4	4	4 核
5	2	4 核
6	0.5	1 核
7	1	1 核
8	4	2 核
9	4	2 核
10	0.5	2 核

以一个云服务器集群管理节点下的 10 个工作节点作为实验对象，使用本文方法中的式(8)和式(9)预测下一个时刻该 10 个工作节点的 CPU 占用率，预测结果如图 3 所示。

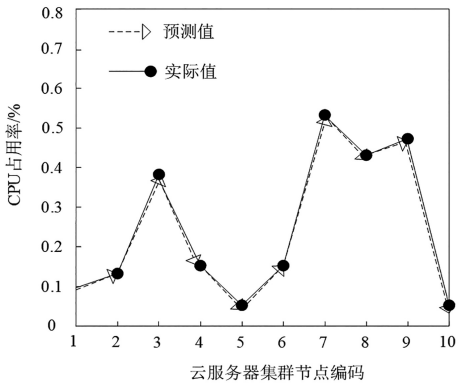

图 3 云服务器集群工作节点 CPU 占用率预测结果

Fig. 3 Prediction result of CPU utilization rate of work nodes in cloud server cluster

从图 3 中可以看出，本文方法对云服务器集群工作节点 CPU 占用率的预测与实际值相当吻合。这表明，我们可以通过预测下一个时刻的 CPU 占用率，对节点资源进行更为精准的调度优化。这一发现不仅证明了本文方法的准确性，更为云服务器的资源管理提供了强有力的支持，有助于实现更为高效、稳定的服务器运行。

在实验环境中选择 5 台主机向云服务器发起数据传输请求指令，使用本文方法对该 5 个集群资源任务进行调度优化，调度优化结果如表 2 所示。

表 2 集群资源任务调度优化结果

Tab. 2 Cluster resource task scheduling optimization results

集群资源任务	执行任务集群节点编码	任务开始时间	任务结束时间
任务 1	1、2、15、17、19	13:05:21	13:09:54
任务 2	8、9、11、20	13:08:17	13:09:25
任务 3	6、10、12、13、14	13:09:01	13:09:44
任务 4	1、2、3、16、19	13:10:24	13:11:51
任务 5	3、4、5、7、18、20	13:06:22	13:09:46

从表 2 的数据分析中，我们可以看到通过本文提出的方法，每个集群资源任务都能得到有效的节点分配，确保了任务执行的流畅性。更为重要的是，在任务执行的时间段内，没有发生节点冲突的情况，这得益于该方法精确的调度能力。当某个节点完成当前任务后，该方法还能为其优化分配其他任务，从而最大化利用集群资源。这些结果有力地证明了本文方法在云服务器集群资源调度优化方面的优秀性能，为提高集群的整体运行效率奠定了坚实基础。

以 10 个云服务器集群节点作为实验对象，使用本文方法，以某个集群资源调度优化任务作为测试环境，使用本文方法对该 10 个云服务器集群节点进行任务调度优化后，以集群节点的任务占用和当前预留空间作为衡量指标，测试结果如图 4 所示。

通过观察图 4,可以清晰地看到,在本文方法下,每个云服务器节点在分配任务后都保留了较大的预留空间。这一设计不仅确保了单个节点的性能优化,更为重要的是,为节点调度其他资源或任务提供了充足的准备。这一特点充分体现了本文方法在云服务器节点调度优化方面的强大能力,为整个集群的高效运行提供了有力支持。

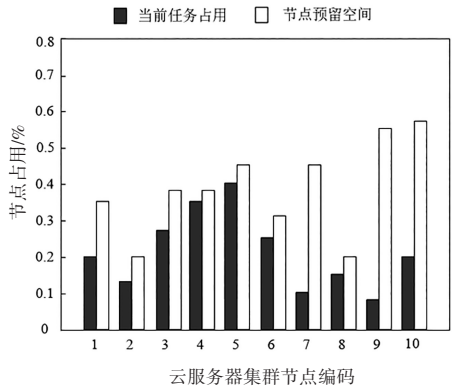

图 4　云服务器集群节点任务占用与预留空间

Fig. 4　Cloud server cluster node task utilization and reserved space

以云服务器内存利用率作为衡量指标,验证本文方法在调度优化云服务器资源任务量不同情况下,云服务器的内存利用率,同时设置内存利用率阈值为 40%,测试结果如图 5 所示。

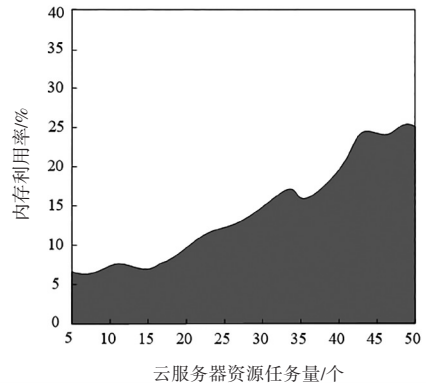

图 5　云服务器内存利用率

Fig. 5　Cloud server memory utilization

根据图 5 的详细分析,本文可以清晰地看到在集群资源任务量较大时,云服务器的内存利用率相应增高。但运用本文的方法进行调度优化时,即便面临大量集群资源任务,仍能确保云服务器的内存利用率维持在一个较低的水平。这意味着,尽管集群任务繁重,但通过本文的方法,云服务器仍然能够并行处理并优化其他集群调度任务。综上所述,本文的方法在云服务器集群资源调度优化方面表现出显著的优势和能力。

3　结论

通过结合Docker容器技术和APS算法,本文对云服务器集群的资源调度优化进行了仿真研究。实验结果表明,该方法能有效提高资源利用率,可有效为云服务器节点分配调度优化任务,实现集群资源的并行处理。然而,如何进一步优化算法以提高响应速度和扩展性仍需深入研究。总体而言,该研究为云服务器的资源调度提供了新的思路和方法,为实际生产环境中的资源管理和优化提供了有益的参考。

参考文献

[1]周晓晶, 谷钰. 基于层次梯度挖掘的数据智能调度算法仿真[J]. 计算机仿真, 2023, 40(4):358-361, 381.
[2]邝祝芳, 陈清林, 李林峰, 等. 基于深度强化学习的多用户边缘计算任务卸载调度与资源分配算法[J]. 计算机学报, 2022, 45(4):812-824.
[3]谢雍生, 黄相恒, 陈宁江. 基于改进 DQN 算法的容器集群自均衡调度策略[J]. 计算机科学, 2023, 50(4):233-240.
[4]KOUSER R R, MANIKANDAN T. A novel clustering and optimal resource scheduling in vehicular cloud networks using MKMA and the CM-CSO algorithm [J]. International Journal of Communication Systems,2023,36(5):18.
[5]UMA J, VIVEKANANDAN P, SHANKAR S. Optimized intellectual resource scheduling using deep reinforcement Q-learning in cloud computing [J]. Transactions on Emerging Telecommunications Technologies,2022,33(5):4463-4482.
[6]刘陈伟, 孙鉴, 雷冰冰, 等. 基于改进粒子群算法的云数据中心能耗优化任务调度策略[J]. 计算机科学, 2023, 50(7):246-253.
[7]赵梓安, 周泓, 雷颖健. 基于改进狼群算法与仿真的单元调度优化[J]. 系统仿真学报, 2022, 34(2): 201-211.
[8]周文操, 张学军, 闫来清. 一种缩小 CCHP 系统优化调度解空间的方法[J]. 太阳能学报, 2023, 44(11): 556-564.
[9]茆书睿, 李熹宁. 基于电池状态的储能电站集群调度策略[J]. 太阳能学报, 2023, 44(8):54-61.
[10]詹成康, 李晓露, 陆一鸣, 等. 基于集群的主动配电网双层分布式优化调度策略研究[J]. 电气传动, 2023, 53(1):81-90.
[11]李亚平, 杨胜春, 毛文博, 等. 基于群体智能的分布式柔性资源有功平衡调度架构及策略[J]. 电力自动化设备, 2022, 42(7):174-182.
[12]胡亚红, 吴寅超, 朱正东, 等. 异构集群节点与作业特性感知资源分配算法[J]. 计算机工程与应用, 2022, 58(18):327-334.
[13]王小雪, 王晓锋, 刘渊. 基于 OpenStack 的高资源利用率 Docker 调度模型[J]. 计算机工程, 2022, 48(9):171-179, 196.
[14]张金波, 梁哲恒, 曾纪钧, 等. 基于优化蚁群算法的软件测试数据自动生成方法[J]. 自动化技术与应用, 2024, 43(11):88-92.
[15]张芳胜, 王妙龄, 季嘉辉, 等. 基于语音识别的多资源组合应急调度指挥系统[J]. 自动化技术与应用, 2024, 43(3):155-159.
[16]董健康, 赵瑛, 张明浩. 基于能耗感知的 Docker 调度机制[J]. 中国科技论文, 2022, 17(11): 1260-1266.
[17]朱文, 胡亚平, 聂涌泉, 等. 基于等级保护的电网调度自动化系统安全防御技术[J]. 电测与仪表,2025,62(7):174-180.
[18]张一帆, 刘章, 朱瑞金, 等. 基于鲁棒优化与阶梯式碳交易的高海拔多能微网群优化调度[J]. 电测与仪表,2025,62(8):29-38.
[19]韦洪波, 阮诗迪, 张雄宝, 等. 基于 Pair Copula 的多风电场风险约束随机经济调度[J]. 电测与仪表,2025,62(7):165-173.
[20]黄瑞, 肖宇, 曾伟杰, 等. Docker 容器下高速总线通信数据实时交互方法[J]. 沈阳工业大学学报, 2022, 44(6):667-671.