苏鸿斌. 基于 Kubernetes 的水平伸缩策略研究与改进[J]. 智能计算机与应用,2025,15(2):59-64. DOI:10.20169/j.issn.2095-2163.250209

基于 Kubernetes 的水平伸缩策略研究与改进

苏鸿斌
(浙江理工大学 计算机科学与技术学院, 杭州 330018)

摘 要: 针对 Kubernetes 中默认水平伸缩策略只支持基于静态指标阈值的响应式伸缩策略,导致容器副本(Pod)在集群负载发生变化时扩缩容操作出现延迟问题,本文提出了改进的水平伸缩策略 DS-HPA。该改进策略由基于负载预测的水平伸缩策略(S-HPA)和基于动态阈值的水平伸缩策略(D-HPA)两部分组成,S-HPA 策略通过三次指数平滑法预测集群未来时刻的负载情况,并调整指数平滑法的平滑指数,使其依据历史数据变化的剧烈程度而动态改变,从而使得该改进策略的预测效果更加贴合集群的负载变化;而 D-HPA 策略是针对负载预测需要积累一定的历史数据量和在预测过程中存在预测间隙无法为集群提供伸缩决策的问题提出的,该策略在预测算法处于无法提供决策的时段,通过动态下降阈值算法提前触发扩缩容操作,以弥补预测算法的不足。实验结果表明,本文提出的改进策略能够较为准确的预测集群的负载变化,使 Pod 副本提前完成扩缩容操作,从而提高集群的响应速度和服务质量。

关键词: Kubernetes; 水平伸缩策略; 负载预测; 指数平滑法; 动态阈值
中图分类号: TP393 **文献标志码**: A **文章编号**: 2095-2163(2025)02-0059-06

Research and improvement of horizontal scaling strategy based on Kubernetes

SU Hongbin
(School of Computer Science and Technology, Zhejiang Sci-Tech University, Hangzhou 330018, China)

Abstract: An improved horizontal scaling strategy DS-HPA is proposed to address the delay issue of scaling operations for Pod replicas in Kubernetes that is caused by the default horizontal scaling strategy only supporting responsive scaling based on static metric thresholds. This improved strategy consists of two parts: the load prediction-based horizontal scaling strategy (S-HPA) and the dynamic threshold-based horizontal scaling strategy (D-HPA). S-HPA predicts the future load of the cluster using triple exponential smoothing and adjusts the smoothing factor based on the severity of historical data changes, thereby improving the prediction accuracy of the strategy. D-HPA addresses the problem of requiring a certain amount of historical data accumulation for load prediction and the inability to provide scaling decisions during prediction gaps, by triggering scaling operations in advance using a dynamic lowering threshold algorithm when the prediction algorithm is unable to provide decisions. Experiments demonstrate that the proposed improved strategy can accurately predict the load changes of the cluster, enabling Pod replicas to complete scaling operations in advance and improving the response speed and service quality of the cluster.

Key words: kubernetes; horizontal scaling strateg; load prediction; exponential smoothing; dynamic threshold

0 引 言

Kubernetes 是最流行的容器编排系统之一,支持多种云平台和操作系统,可以帮助用户自动化管理、调度和扩展容器化应用程序。通过使用 Kubernetes 容器编排系统,用户可以更加便捷地部署和管理容器化的应用程序,从而极大地提升了云计算的效率和容器技术的可靠性[1-2]。Kubernetes 容器编排系统提供了强大的资源调度和管理功能,使用户能够轻松地分配和管理集群中的资源[3]。水平伸缩控制技术是 Kubernetes 的核心技术之一,但是由于默认的伸缩策略属于静态调度策略,随着

基金项目: 浙江省重点研发计划项目(2020C03094); 浙江省教育厅一般科研项目(Y202147659); 浙江省教育厅项目(Y202250706); 浙江省基础公益研究计划项目(QY19E050003)。
作者简介: 苏鸿斌(1997—),男,硕士研究生,主要研究方向:计算机网络与分布式处理。Email:908377408@qq.com。
收稿日期: 2023-08-25

云计算的普及和应用负载的不断增长,该默认策略已经不能满足现代企业对高效、弹性、可扩展系统的需求[4]。张启辉等[5]针对 Kubernetes 默认伸缩策略只考虑了容器副本(Pod)的基本资源信息而忽略了节点的实时资源数据,导致缩容效果不理想的问题,提出了一种改进的 Kubernetes 动态缩容模型,该模型综合考虑了 Pod 和节点的资源利用率、Pod 的重要性、节点的负载均衡等因素,通过一个综合评分函数来确定需要删除的 Pod,有效提升了集群的负载均衡情况;沐磊等[6]提出了一种改进的弹性伸缩策略,实现了对集群综合负载的预测,解决了默认伸缩策略的衡量指标单一和响应延迟问题,使得集群的负载处于良好的均衡状态;Toka 等[7]提出了一种使用自适应人工智能技术实现的 Kubernetes 自动缩放策略,利用机器学习算法预测负载,并根据预测结果动态地调整 Kubernetes 的资源分配,从而实现自适应的扩展和收缩;Ding 等[8]提出了一种新颖的组合伸缩策略,基于优化和预测的联合自动扩缩策略(Combined Optimization - based and Predictive Autoscaling,COPA),可以根据实时的工作负载、期望的响应时间提供微服务实例方案,利用排队网络模型来计算一个既能最小化违约成本又能最小化资源成本的组合伸缩策略,实现了对 Pod 的水平伸缩和垂直伸缩的协同优化,提高了集群的资源利用率和服务质量。

针对 Kubernetes 的伸缩技术优化研究可以看出,对 Kubernetes 伸缩技术的研究主要分为基于动态阈值的水平伸缩策略和基于负载预测的水平伸缩策略两种,两种策略都有各自的优缺点,可以形成互补,本文通过结合两种伸缩策略,使得 Kubernetes 集群在运行过程中动态切换采用不同的伸缩策略,最大程度的解决扩缩操作延迟问题。

1 Kubernetes 默认伸缩策略

1.1 Kubernetes 水平伸缩工作原理

水平容器副本自动伸缩(Horizontal Pod Autoscaler,HPA)控制器的工作原理如图 1 所示。具体伸缩流程如下:

步骤 1 Kubernetes 使用监控服务器(Metric Server)收集容器的指标数据;

步骤 2 HPA 控制器通过接口服务器向监控服务器获取相关资源指标数据,如 CPU 利用率,并计算相关节点的平均 CPU 利用率;

步骤 3 一旦 HPA 确定当前负载情况不处于预期值,将决策是否需要添加或删除 Pod,随后 HPA 通过计算得到扩缩容期望值,并将结果发送给部署控制器;

步骤 4 部署控制器会持续监控副本集控制器和 Pod 的状态,如果发现与期望状态不一致,就会向副本集控制器发送调整请求;

步骤 5 最后由副本集控制器向节点代理器发送扩缩操作命令。

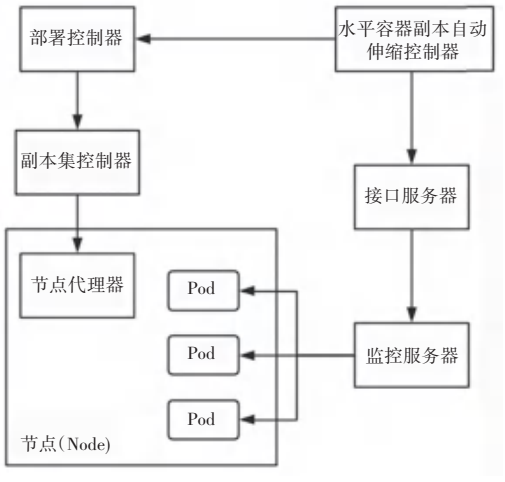

图 1 HPA 工作原理

Fig. 1 Working principle of HPA

1.2 水平伸缩实现原理

Kubernetes 的水平容器副本自动伸缩策略(HPA)是一种自动扩展或缩小容器副本数量的策略,能够根据应用程序的需求和负载情况来自动地调整 Pod 的数量,从而确保应用程序的高可用性和性能[9]。

HPA 的工作原理:当 Pod 的 CPU 利用率或其他指标超出预设的范围时,Kubernetes 会自动增加 Pod 的数量;当 Pod 的 CPU 利用率或其他指标低于预设的范围时,Kubernetes 会自动减少 Pod 的数量。这种自动化的扩展和缩小操作可以确保应用程序具有更好的可用性,同时避免了手动管理和调整容器数量的繁琐操作和耗时[10]。

扩缩容计算公式如下:

$$dR = ceil[cR \times (\frac{cM}{dM})] \tag{1}$$

其中,cM 表示当前的值;dM 表示期望的值;cR 表示当前的 Pod 副本数;dR 表示希望扩缩容后的 Pod 副本数。

扩缩容的决策流程:首先,通过监控服务器获取关键指标,记为 cM;其次,获取用户定义的预期值 dM,计算 Pod 数量;最后,将期望的 Pod 数量与实际

Pod 数量进行比较，向控制器提交伸缩操作决策，由控制器完成伸缩。

1.3　存在的问题和研究方向

容器云常用的伸缩策略主要分为两种：基于目标资源指标阈值的响应式伸缩策略和基于负载预测的伸缩策略[11]。

1）基于目标资源指标阈值的响应式伸缩策略

基于目标资源指标阈值的响应式伸缩策略是将周期性监控目标的资源指标数据与所设定的指标阈值进行对比，一旦超过相应指标阈值就会触发扩缩容机制，从而动态的改变相关节点的 Pod 副本数量。该策略能够对突发流量迅速做出响应，但是无法预测集群未来时刻的资源需求量，因此相应的自动伸缩调整往往都发生在集群的负载情况已经变化后。因为相应的扩缩容操作存在一定的延迟，导致集群的服务质量降低。

2）基于负载预测的伸缩策略

基于负载预测的伸缩策略通过对系统的历史性能和资源配置数据的分析，预测未来的容量需求，相较于基于阈值的响应式伸缩策略，能够提前响应突发情况，使集群的弹性伸缩变化更加平滑。但是负载预测难免会因为前期历史数据不足导致预测结果出现误差，从而导致扩容不足或者过度缩容的情况，以至于引起服务质量的降低。而且基于负载预测的伸缩策略在预测期间是无法对集群进行伸缩操作的，存在一定的真空期。

本文首先提出基于负载预测的水平伸缩策略 S-HPA，能够根据历史数据对未来的容量进行评估预测，有效地预测 Kubernetes 集群未来时刻的资源需求量；其次，针对负载预测算法存在预测间隙无法及时响应的问题，提出基于动态阈值的水平伸缩策略 D-HPA，有效降低负载预测算法带来的负面影响；最后，将两种伸缩策略应用于 Kubernetes 集群中，并设置相关参数，使得集群在运行过程中动态切换不同的伸缩策略。

2　改进的水平伸缩策略设计

2.1　基于负载预测的水平伸缩策略设计

Kubernetes 集群的负载预测伸缩策略通常有两种实现方式：一种是基于机器学习算法实现的伸缩策略；另一种则是基于时间序列算法实现的伸缩策略。但是基于机器学习算法的策略通常都需要长时间以及大量数据来进行模型训练，对于 Kubernetes 集群来说其适用性不强，因此本文采用时间序列算法中的三次指数平滑法进行负载预测。

2.1.1　指数平滑法

指数平滑法是一种时间序列预测方法，用于预测未来的趋势，基于对历史时序数据加权平均的思想实现，距离当前时刻越近的数据点被给予越大的权重[12-14]。指数平滑法通过计算当前值与先前预测值之间的加权平均来预测下一个时期的值。加权系数取决于选择的平滑常数，该常数控制着过去数据点对预测的贡献程度。

指数平滑法有多种变体，包括简单指数平滑法、双指数平滑法和三指数平滑法等。这些方法可以根据需要进行调整，以适应不同的预测问题。简单指数平滑法和多次指数平滑法的公式如下：

$$S_t = \alpha x_t + (1-\alpha) S_{t-1} \tag{2}$$

$$S_t^{(n)} = \alpha S_t^{(n-1)} + (1-\alpha) S_{t-1}^{(n)} \tag{3}$$

其中，S_t 为 t 时刻的平滑值；α 为平滑指数且 $\alpha \in [0,1]$；x_t 为历史数据序列。

由于 Kubernetes 集群的负载变化并不是线性的，采用简单指数平滑法和双指数平滑法得出的预测结果总是出现滞后误差的问题，因此本文采用三次指数平滑法。三次指数平滑法是在双指数平滑法的基础上再做单指数平滑的方法，不能单独进行预测，必须与双指数平滑法和单指数平滑法配合。

2.1.2　基于三次指数平滑法的伸缩策略实现

假设 $y_t, y_{t-1}, \cdots, y_{t-n+1}$ 为 Kubernetes 集群 t 时刻收集的 n 次历史数据序列，此时在 t 时刻且历史数据量为 n 的时刻单指数平滑法的公式如下式所示：

$$S^{(1)} = \alpha y_t + (1-\alpha) S_{t-1}^{(1)} \tag{4}$$

在得到单指数平滑法预测值的基础上根据式(3)进行指数平滑，得到双指数平滑法和三次指数平滑法的预测值，计算公式如下：

$$\begin{cases} S^{(2)} = \alpha S_t^{(1)} + (1-\alpha) S_{t-1}^{(2)} \\ S^{(3)} = \alpha S_t^{(2)} + (1-\alpha) S_{t-1}^{(3)} \end{cases} \tag{5}$$

则在 $t+T$ 时刻的预测值可用下式计算：

$$\hat{Y}_{t+T} = a_t + b_t \times T + c_t \times T^2 \tag{6}$$

其中，T 为预测步长。

$$\begin{cases} a_t = 3S_t^{(1)} - 3S_t^{(2)} + S_t^{(3)} \\ b_t = \dfrac{\alpha}{2(1-\alpha)^2}\left[(6-5\alpha)s_t^{(1)} - 2(5-4\alpha)s_t^{(2)} + (4-3\alpha)s_t^{(3)}\right] \\ c_t = \dfrac{\alpha}{2(1-\alpha)^2}\left[S_t^{(1)} - 2S_t^{(2)} + S_t^{(3)}\right] \end{cases} \tag{7}$$

因此基于三次指数平滑法对集群在 $t+T$ 时刻

的负载预测值可用下式计算：

$$F_{t+T} = 3S_t^{(1)} - 3S_t^{(2)} + S_t^{(3)} + \frac{\alpha}{2(1-\alpha)^2}[(6-5\alpha)S_t^{(1)} - 2(5-4\alpha)S_t^{(2)} + (4-3\alpha)S_t^{(3)}] \times T + \frac{\alpha}{2(1-\alpha)^2}[S_t^{(1)} - 2S_t^{(2)} + S_t^{(3)}] \times T^2 \quad (8)$$

S-HPA 采用基于时间序列分析的方法，使用三次指数平滑法对 Kubernetes 集群的 Pod 副本数进行负载预测，能够有效地利用历史时序数据及其趋势性信息，对未来时刻的 Pod 副本数进行精确的预测，并且对突发性负载变化有较好的适应性和鲁棒性。三次指数平滑法通过对历史数据进行加权平均，将趋势性、季节性和随机性因素分离，并根据历史趋势对未来趋势进行预测，从而实现对未来时刻 Pod 副本数的准确预测，预测滞后的问题得到了改善。但是由于指数平滑法的平滑指数为静态值，导致预测结果并不能时刻契合集群的负载变化情况，仍会存在预测偏差。

2.1.3 三次指数平滑法的改进

在使用三次指数平滑法进行数据预测时，平滑指数(Smoothing Factor)的选择应该考虑到历史数据的变化程度和预测的准确性。王长江等[15]研究可知，如果历史数据的变化比较剧烈，平滑指数应该设置得比较小，以便更快地适应新的趋势；而如果历史数据的变化比较缓慢，平滑指数应该设置得比较大，以便更稳定地跟踪趋势。由于集群的负载变化快慢情况是随时间的变化而变化的，在凌晨的时刻负载都是处于较低且变化不大的状态；正常的工作时间，负载会开始剧烈的起伏。为了预测伸缩策略的预测值能够更贴近负载变化情况，本文将对三次指数平滑法的平滑指数进行改进。

由于平滑指数的选择与历史数据的变化剧烈程度成反比，引入标准差 σ 用于评判历史数据的变化剧烈程度，如下式所示：

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \mu)^2}{n}} \quad (9)$$

其中，x_i 表示第 i 个历史数据值，μ 表示所有历史数据的平均值。

因此 σ 越大，表示历史数据之间差异越大；σ 越小，表示历史数据之间差异越小。平滑指数 α 的取值范围一般为(0,1)，α 的计算公式如下：

$$\alpha = 1 - \frac{|\hat{X} - \mu|}{\sigma} \quad (10)$$

其中，$\hat{X}$ 为最近 5 次的数据平均值。

α 将由标准差 σ 和最近的 5 次数据值的平均值决定。带入最近 5 次数据的平均值用于加强近期的数据变化程度对 α 的影响。

2.2 基于动态阈值的水平伸缩策略设计

Kubernetes 集群采用基于负载预测的水平伸缩策略时，集群通过对历史数据的分析和预测，可以更加准确地判断未来负载情况，从而更好地扩容或缩容，提高系统的灵活性、可靠性和稳定性，使得系统能够更好地适应变化的负载需求。但由于该预测策略需要历史数据的积累，存在预测间隙，Kubernetes 集群在这两段时间内依然会使用默认的水平伸缩策略，导致集群存在扩缩操作延迟的问题。因此本文提出一种基于动态下降阈值的响应式伸缩策略，用于缓解集群处于历史数据积累期间和预测间隙时可能出现的扩缩容操作延迟问题。对默认基于目标资源指标阈值的响应式伸缩策略进行改进，在集群负载上升阶段对阈值进行动态调整，适当下调扩容阈值，使得集群能够更提前触发扩容机制，从而更加从容的应对负载上升的情况。而在 Kubernetes 集群中的流量处于下降阶段时，D-HPA 会动态下调缩容阈值，这种缩容阈值的下调能够有效延迟缩容机制的触发，以防止频繁扩缩容的情况。

动态下降阈值算法通过获取 Kubernetes 集群中 Pod 副本的 CPU 利用率来计算。$C = \{x_1, x_2, \cdots, x_n\}$ 为 t 时刻采集的 n 次 CPU 历史利用率，则 CPU 在 t 时刻的增长量可以由下式计算得到。

$$f(x) = x_n - x_{n-1} \quad (11)$$

则扩缩容的阈值动态下调公式如下：

$$T_{\text{down}} = T - \left| \frac{\sum_{i=1}^{n} x_i}{n} \times f'(x) \right| \quad (12)$$

其中，T_{down} 为新的阈值；T 为当前阈值；$f'(x)$ 为当前集群中的 CPU 增长率。

通过动态调整基于目标资源指标的响应式伸缩策略的阈值，使其集群能够在负载发生变化前执行本应的伸缩策略，从而缓解静态的响应式伸缩策略存在的扩缩容操作延迟问题。

2.3 改进伸缩策略的伸缩流程

DS-HPA 策略是一种综合了负载预测和动态阈值的水平伸缩策略，能够通过对历史数据的分析和

对未来负载的预测，更加准确地预测 Kubernetes 集群的资源需求，从而实现更加智能化的扩缩容操作。

DS-HPA 策略实现流程：

步骤 1　判断集群是否处于预测期间，是则跳转至步骤四，否则跳转至步骤二；

步骤 2　判断集群记录的历史数据量是否大于等于 20，是则跳转至步骤三，否则跳转至步骤四；

步骤 3　调用基于负载预测的伸缩策略，计算未来时刻期望值并跳转至步骤五；

步骤 4　调用基于动态阈值的伸缩策略进行计算期望值；

步骤 5　根据得到的期望值进行相应扩缩容操作。

3　实验设计和结果分析

3.1　实验设计

通过在 6 台虚拟机上部署 Kubernetes 节点组成集群的形式安装于 PC 机上，Kubernetes 集群中包含 1 个主节点(Master)和 5 个从节点(Node)。集群的 6 个节点的硬件配置信息见表 1。

表 1　6 个节点硬件配置信息

Table 1　Hardware configuration information of six nodes

节点名称	CPU 核心数	运行内存/GB	网络带宽/Mbps	磁盘容量/GB
Master	2	4	100	10
Node0	4	6	100	20
Node1	2	4	50	20
Node2	2	4	100	10
Node3	2	4	100	10
Node4	2	2	100	10

节点的软件配置信息见表 2。

表 2　节点的软件配置信息

Table 2　Software configuration information for nodes

软件名称	版本
操作系统	CentOS 7.8
Docker	18.06
Kubernetes	1.19

Pod 是 Kubernetes 中的最小调度单元，可以承载一个或多个容器，用于响应和处理用户请求。为模拟真实生产环境，Pod 采用服务器常用的 Web 应用。

根据 Toka 等[14]对 Facebook 网站进行流量追踪所采集的数据进行分析可知，服务器的负载随时间的变化而变化。在晚上 8 点至第二天早上 5 点期间，服务器的负载量一直处于低水平状态，早上 5 点后负载量开始增加，并于中午 11 点到达顶峰。由此可知服务器的负载量符合标准的正态分布，为了模拟服务器真实的负载情况，本文实验将结合标准正态分布函数向集群发送任务请求。设 $f(x)$ 为集群在 x 时刻收到的任务请求量，任务请求数量公式如下：

$$f(x)=\frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{(x-\mu)^2}{2\sigma^2}}\times 5\,000+20 \tag{13}$$

其中，x 取值范围为 [2 - 22]，代表 24 小时制时间 2 点到 22 点，参数 μ 和 σ 取值如下式所示：

$$\begin{cases}\mu=11\\ \sigma=\dfrac{5}{2\sqrt{2\log(2)}}\end{cases} \tag{14}$$

随时间变化，集群收到的请求量变化图如图 2 所示。

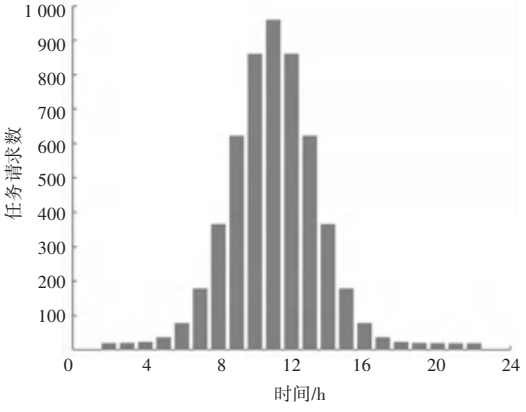

图 2　集群请求量变化图

Fig. 2　Cluster request quantity variation

3.2　结果分析

本组实验中集群采用的伸缩策略分别是 Kubernetes 的默认伸缩策略、基于三次指数平滑法的伸缩策略和本文设计的 DS-HPA 伸缩策略。其中每种伸缩策略都进行 3 次实验，实验途中记录 Pod 副本数及任务完成时间，

将 3 次实验记录的数据计算平均值，得到最终实验结果。

随集群的请求量变化，不同伸缩策略的 Pod 水平伸缩对比图如图 3 所示，可以看出本文提出的基于负载预测和动态阈值的水平伸缩策略 DS-HPA 和采用三次指数平滑法的伸缩策略在负载量呈现上升趋势时均能够提前执行扩容操作；当负载量呈现下降状态时，Pod 也能够及时进行缩容操作，与默认的 Pod 水平伸缩策略相比具有更好的拟合服务器负

载量变化的特性。由于采用三次指数平滑法的伸缩策略的平滑指数为固定值,无法根据集群负载变化的剧烈程度动态调整,导致在集群负载量变化时频繁进行扩缩容操作,频繁的扩缩操作将导致集群的运算资源浪费。

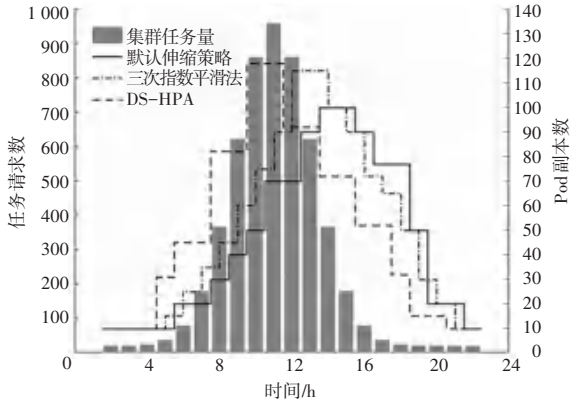

图 3 Pod 水平伸缩对比图

Fig. 3 Comparison of Pod horizontal scaling

不同伸缩策略的任务完成时间对比如图 4 所示,可以看出随着负载的提升,采用 DS-HPA 的集群任务完成时间明显低于采用另外两种伸缩策略的集群,且任务完成时间变化的幅度也比较平缓。采用三次指数平滑法的集群虽然也进行了提前的扩容操作,但是由于操作过于频繁,浪费了大量的资源和时间,最终导致任务完成时间高于采用 DS-HPA 策略的集群,且任务完成时间的曲线变化不平缓,不利于集群的稳定运行。在集群的负载下降阶段,由于采用默认调度策略的集群堆积了较多任务,导致完成时间的下降,较另外两种策略的集群出现了滞后现象。

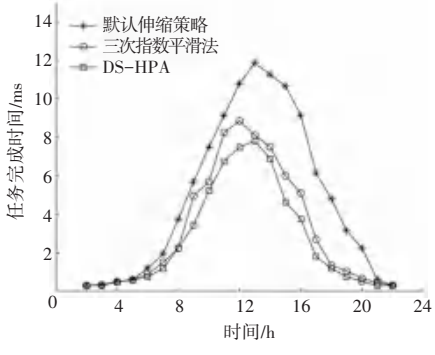

图 4 任务完成时间对比图

Fig. 4 Comparison of task completion times

综合对比,本文提出的 DS-HPA 策略能够使服务器的整体负载均衡度处于更加良好的状态,从而提高集群的服务质量。

4 结束语

流量快速变化的情况下,Kubernetes 的扩缩容延迟会导致大量服务无法及时处理。为了解决这个问题,本文结合基于负载预测和基于动态阈值两种伸缩策略的优点,提出了 DS-HPA 伸缩策略。在预测时使用改进的三次指数平滑法进行负载预测,在历史数据量不足或处于预测期间时使用基于动态阈值的响应式伸缩策略。DS-HPA 策略能够预测未来时刻容器云的资源需求,同时解决了预测算法带来的不足之处,提高集群的响应速度和服务质量。

参考文献

[1] 武志学. 云计算虚拟化技术的发展与趋势[J]. 计算机应用,2017,37(4):915-923.

[2] KOZHIRBAYEV Z, SINNOTT R O. A performance comparison of container-based technologies for the cloud[J]. Future Generation Computer Systems, 2017(68): 175-182.

[3] KUBERNETES T. Kubernetes[J]. Kubernetes. Retrieved May, 2019(24): 2019.

[4] LUKSA M. Kubernetes in action 中文版[M]. 北京:电子工业出版社, 2019.

[5] 张启辉,未来. 一种改进的 Kubernetes 动态缩容模型[J]. 数据通信,2019(5):38-42.

[6] 沐磊,李洪赭,李赛飞. 一种改进的 Kubernetes 弹性伸缩策略[J]. 计算机与数字工程,2022,50(2):327-331.

[7] TOKA L, DOBREFF G, FODOR B, et al. Adaptive AI-based auto-scaling for Kubernetes[C]//Proceedings of the 20th IEEE International Symposium on Cluster, Cloud and Internet Computing. Piscataway,NJ:IEEE, 2020: 599-608.

[8] DING Z, HUANG Q. COPA: A combined autoscaling method for kubernetes[C]// Proceedings of the 2021 IEEE International Conference on Web Services. Piscataway,NJ:IEEE, 2021: 416-425.

[9] 阿里云. 弹性伸缩[OL]. (2020-04-05). https://www.aliyun.com/product/ess.

[10] HUO Q, LI C, LI S, et al. High concurrency response strategy based on Kubernetes horizontal pod autoscaler[J]. Journal of Physics: Conference Series, 2023, 2451(1): 012001.

[11] 单朋荣,杨美红,赵志刚,等. 基于 Kubernetes 云平台的弹性伸缩方案设计与实现[J]. 计算机工程,2021,47(1):312-320.

[12] GOBALAKRISHNAN N, ARUN C. SIS: A scheme for dynamic independent task scheduling in a cloud environment[C]// Proceedings of the 2016 International Conference on Control, Instrumentation, Communication and Computational Technologies. Piscataway,NJ: IEEE, 2016: 272-277.

[13] 杨博帆,张琳,张搏,等. 动态多模型指数平滑法融合的在线预测方法[J]. 系统工程与电子技术,2020,42(9):2013-2021.

[14] TOKA L, DOBREFF G, FODOR B, et al. Machine learning-based scaling management for Kubernetes edge clusters[J]. IEEE Transactions on Network and Service Management, 2021, 18(1): 958-972.

[15] 王长江. 指数平滑法中平滑系数的选择研究[J]. 中北大学学报(自然科学版),2006(6):558-561.