面向多等级应用的气象云资源调度方法研究

梁中军[1,2],孙志于[1],韩同欣[2],宋雅婷[1],张正阳[1]
(1. 新疆气象信息中心,新疆 乌鲁木齐 830002;
2. 国家气象信息中心,北京 100081)

摘 要:气象云平台是集约支撑气象业务发展基础资源平台,如何合理调度气象云平台的基础资源已成为气象业务稳定运行的关键。由于不同等级气象应用在汛期和日常具有特殊支撑需求,以及部分功能组件对气象云平台的物理服务器有约束要求,现有调度算法难以被直接应用。为此,该文对问题建模,设计目标函数分别评价气象云平台的资源利用水平和对重要核心业务的支撑情况,基于物理服务器与功能组件之间的约束关系,将问题建模成一个多约束多目标优化问题。然后,设计一种改进蛙跳算法的气象云资源调度方法求解模型,该方法利用适应度函数来综合评价模型目标,并重新定义蛙跳算子,利用局部最优交叉操作和最优青蛙变异策略来迭代搜索最优资源调度方案。最后,通过实验验证了该方法的效果。

关键词:气象云;资源调度;多等级应用;蛙跳算法;多目标优化
中图分类号:TP393 **文献标识码**:A **文章编号**:1673-629X(2022)08-0203-07
doi:10.3969/j.issn.1673-629X.2022.08.033

Research on Resource Scheduling in Meteorological Cloud Environment for Multi-class Application

LIANG Zhong-jun[1,2], SUN Zhi-yu[1], HAN Tong-xin[2], SONG Ya-ting[1], ZHANG Zheng-yang[1]
(1. Xinjiang Meteorological Information Center, Urumqi 830002, China;
2. National Meteorological Information Center, Beijing 100081, China)

Abstract: Meteorological cloud platform is a basic resource platform that supports the development of meteorological services intensively. How to reasonably schedule the basic resources of meteorological cloud platform has become the key to the stable operation of meteorological services. Because different levels of meteorological applications have special support requirements in flood season and daily life, and some functional components have constraints on physical servers of meteorological cloud platform, existing scheduling algorithms are difficult to be directly applied. Therefore, we model the problem and design objective functions to evaluate respectively the resource utilization level of meteorological cloud platform and its support for important core business. Based on the constraint relationship between physical server and functional components, the problem is modeled as a multi-constraint multi-objective optimization problem. Then, a weather cloud resource scheduling model with improved leaping frog algorithm is designed. The fitness function is used to comprehensively evaluate the model objective, and the leaping frog operator is redefined, and the local optimal crossover operation and the optimal frog mutation strategy are used to iteratively search for the optimal resource scheduling scheme. Finally, the effectiveness of the proposed method is verified by experiments.

Key words: meteorological cloud; resource scheduling; multi-class application; leaping frog algorithm; multi objective optimization

0 引 言

随着云计算、大数据技术的不断发展,气象云平台[1]已成为支撑天气、气候、公共气象服务等业务的重要平台。其中,如何合理分配气象云平台中的基础资源,已成为气象业务稳定运行的关键。

当前,研究者针对不同场景[2-4]的云平台调度问题,从任务指派[5-9]、虚拟资源调度[10-14]和物理资源调度[15-21]等方面提出了各类启发式方法。例如,文献[6]提出了一种基于强化学习的多目标任务调度算法;文献[7]提出了一种考虑时间、成本、CPU、内存和

收稿日期:2022-03-07 **修回日期**:2022-07-08
基金项目:国家重点研发计划课题(2018YFC1507102,2019YFA0606904)
作者简介:梁中军(1983-),男,博士,高级工程师,CCF 会员(H9821M),研究方向为云计算、气象大数据管理与数据治理;通信作者:孙志于(1973-),男,高级工程师,研究方向为云计算、气象大数据管理与应用。

带宽等多维约束的任务调度算法;文献[8]从任务调度角度对其调度流程、主要算法和评测指标进行了综述;文献[12]提出了一种新型的面向网络性能优化的计算资源分配及调度机制;文献[17]针对业务之间存在的复杂约束关系,提出了一种面向私有云的业务迁移部署方法。虽然现有研究针对公有云、私有云等场景,从负载均衡[9]、节能提升利用率[13-14,20-21]、低成本高 QoS[18]等角度给出了大量的算法,然而,与上述研究场景不同,以上方法没有针对气象业务应用的特点设计,无法直接应用在气象云平台里,同时,现有算法没有考虑部分功能组件对云平台的物理服务器有不兼容的情况,不能满足气象云平台的资源调度需求。

与传统云平台中的业务应用不同,气象业务应用具有更明显的周期性特点,在日常气象业务应用对基础设施资源的需求相对平稳,气象云平台可以追求较高的资源利用水平。在汛期气象业务的重要核心组件需要尽可能预留充足的基础设施资源以应对重大天气过程、重大气象保障等对气象监测应用、气象预报应用、灾害预警应用的业务要求。为此,气象云平台不仅需要考虑平台资源的集约性,同时要确保重要核心应用在平台运行的稳定性。

为解决上述问题,该文对气象云平台的资源调度问题进行建模,设计一种改进蛙跳算法的气象云资源调度方法,并通过实验验证了方法的效果。主要贡献体现在以下四个方面:

(1)根据气象业务对基础设施资源需求的特点,设计两个目标函数来分别评价气象云平台的资源利用水平和对重要核心业务的支撑情况,基于物理服务器与功能组件之间的约束关系,将问题建模成一个多约束多目标优化问题。

(2)根据气象业务在日常和汛期的资源需求特点,设计适应度函数来综合衡量模型目标,将问题转化为多约束单目标问题。

(3)设计一种改进蛙跳算法的气象云平台资源调度方法。方法将资源调度方案映射成为蛙跳算法中的石头位置,重新定义蛙跳算子,利用局部最优交叉操作和最优青蛙变异策略来提升方法的迭代搜索能力。

(4)通过实验仿真验证了方法的效果。

1 气象云平台资源调度场景描述

为支撑天气、气候、公共气象服务等业务对基础设施资源的需求,气象云平台使用虚拟机和物理服务器来承载不同业务系统的功能组件,为其提供 CPU、内存、I/O、存储等资源。由于气象业务系统对软硬件环境有一定要求,且部分系统的组件之间有部署要求,功能组件在部署前就需要满足多种约束关系。主要包括:服务器的稳定性约束和功能组件部署约束。其中,前者是为了保障气象云平台的承载业务不因资源抢占而导致中断,后者主要满足业务运行的前置条件。由于气象业务应用在日常和汛期具有不同的特点,且业务迁移成本较高,资源调度通常在汛期前后进行开展。具体如图 1 所示。

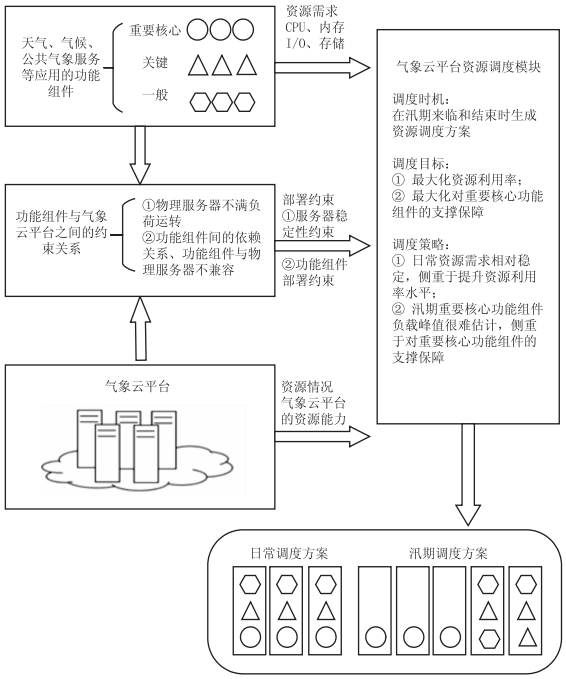

图 1 气象云平台资源调度场景

由于气象云平台中的基础设施资源有限,为满足业务的有序发展,气象应用被划分成若干等级,分级提供基础资源支撑。根据气象业务的重要程度将功能组件划分成若干等级(如:重要核心、关键和一般)。根据功能组件对气象云平台的资源需求(CPU、内存、I/O、存储等),以及上述两种约束关系对功能组件进行动态部署。其中,在日常资源需求稳定时,侧重提升资源利用率,以提升气象业务的集约化水平。在汛期重要核心功能组件的峰值很难估计,侧重于保障重要核心功能组件的稳定运行(即,为高等级应用提供更多基础设施资源)。

2 气象云平台资源调度的问题建模

该文将面向多等级应用的气象云资源调度问题建模如下:

已知有 l 个等级的 w 个气象业务功能组件需要动态部署在气象云平台上。其中云平台由 n 台物理服务器组成,资源能力为 $\{T^1,\cdots,T^k,\cdots,T^n\}$,$T^k=\{t_1^k,t_2^k,\cdots,t_r^k\}$ 为第 k 台物理服务器在 r 种资源上的能力,$\{Q^1,Q^2,\cdots,Q^i,\cdots,Q^w\}$ 为 w 个气象业务功能组件当前的资源需求,$Q^i=\{q_1^i,q_2^i,\cdots,q_r^i\}$ 为第 i 个功能组件的 r 种资源需求。假设气象云平台承载的重要核心功能组件

共有 u 个,其标识为 $G=\{g_1,g_2,\cdots,g_u\}$ 且满足 $g_u\in[1,w]$。面向多等级应用的气象云平台资源调度问题就是在满足约束下计算一种资源分配方法,将所有功能组件映射到气象云平台的物理服务器上,实现日常和汛期的业务目标。

基于以上描述,该问题模型的主要参数及数据符号标识如表 1 所示。

表 1　模型的主要参数及数据符号标识

主要参数	数学符号
功能组件总数	w
物理服务器总数	n
功能组件的等级数	l
功能组件的等级	$\{L^1,\cdots,L^i,\cdots,L^w\}$, $L^i\in[0,l]$
功能组件的资源需求	$\{Q^1,\cdots,Q^i,\cdots,Q^w\}$, $Q^i=\{q_1^i,q_2^i,\cdots,q_r^i\}$
气象云平台的资源能力	$\{T^1,\cdots,T^k,\cdots,T^n\}$, $T^k=\{t_1^k,t_2^k,\cdots,t_r^k\}$
不同等级功能组件对资源的预留要求	$\{S^1,\cdots,S^i,\cdots,S^l\}$
功能组件部署的依赖约束关系	$D=\{d_1,d_2,\cdots,d_o\}$
功能组件部署的冲突容约束关系	$C=\{c_1,c_2,\cdots,c_h\}$
重要核心功能组件的标识	$G=\{g_1,g_2,\cdots,g_u\}$
资源调度方案	X

气象云平台资源调度问题模型如下:

目标一:最大化气象云平台的资源利用率,具体如公式(1)所示。

$$\max\prod_{a=1}^{r}\frac{\sum_{k=1}^{n}\sum_{i=1}^{w}(x_{ik}\times q_a^i)}{\sum_{k=1}^{n}(y_k\times t_a^k)} \tag{1}$$

式中,$y_k=\begin{cases}1,\sum_{i=1}^{n}x_{ik}\geqslant 1\\0,\sum_{i=1}^{n}x_{ik}=0\end{cases}$,$x_{ik}$ 表示功能组件 i 与物理服务器 k 之间的调度关系,取值为 1 时表示 i 在 k 上部署。

目标二:最大化气象云平台对重要核心功能组件的支撑保障,即对重要核心业务预留最大基础设施资源,具体如公式(2)所示。

$$\max\prod_{a=1}^{n}F(y_k) \tag{2}$$

式中,

$$F(y_k)=\begin{cases}\prod_{a=1}^{r}\dfrac{(t_a^k-\sum_{i=1}^{w}(x_{ik}\times q_a^i))}{t_a^k\times\sum_{i=1}^{w}x_{ik}}, & \sum_{i\in G}x_{ik}\geqslant 1\\1, & \sum_{i\in G}x_{ik}=0\end{cases}$$

G 为重要核心功能组件的标识,x_{ik} 表示功能组件 i 与物理服务器 k 之间的调度关系。

约束条件一:服务器的稳定性约束。

为保证气象业务能够稳定运转,承载功能组件的物理服务器需要预留一定资源,保证气象云平台不会满负荷运行(即:物理服务器的基础能力要大于其所承载应用的需求之和)。资源预留将根据功能组件的等级有所差异,高等级预留更多资源。假设功能组件有 l 个等级,则 $\{S^1,\cdots,S^i,\cdots,S^l\}$ 分别是承载不同等级功能组件所对应物理服务器需预留的资源,其中 $S^i=\{s_1^i,s_2^i,\cdots,s_r^i\}$。该约束条件可由公式(3)表示。

$$\forall a\in[1,r],\sum_{i=1}^{n}x_{ik}\times q_a^i<t_a^k-s_a^{B_k} \tag{3}$$

式中,B_k 是物理服务器 k 上承载功能组件的最高等级。即物理服务器要满足其承载的最高等级功能组件的资源预留要求。

约束条件二:功能组件部署约束。

由于气象业务系统对软硬件环境有一定要求,部分功能组件之间存在依赖关系,资源调度需要满足一定前置条件。假设功能组件部署有 o 条依赖约束 $D=\{d_1,d_2,\cdots,d_o\}$ 和 h 条冲突约束 $C=\{c_1,c_2,\cdots,c_h\}$,则约束可由下列公式表示。

$$\forall c_h\in C\wedge c_h=<e,F,0>\wedge k\in F,\ x_{ek}=0 \tag{4}$$

$$\forall d_o\in D\wedge d_o=<a,b,1>,x_{ak}\times x_{bk}=1 \tag{5}$$

其中,任意冲突约束 $c_h=<e,F,0>$ 表示物理服务器 F 不满足功能组件 e 的部署条件;任意依赖约束 $d_o=<a,b,1>$ 表示功能组件 a 和 b 需要部署在同一台物理服务器上。

3　基于改进蛙跳算法的气象云平台资源调度方法

为解决气象云平台资源调度问题,该文设计改进蛙跳算法来求解模型。由于面向多等级应用的气象云平台资源调度问题考虑因素较多,同时传统资源调度问题本身属于 NP-Hard 问题,因此难以精确求解问题。由于蛙跳算法在求解多约束问题上具有搜索能力强、与本问题求解更易映射的特点,该文以蛙跳算法框架为基础,将问题求解与蛙跳算法进行映射,重新定义蛙跳算子,设计适应度函数,利用局部最优交叉操作和

最优青蛙变异策略来迭代搜索最优方案。

3.1 气象云平台资源调度方案映射

为迭代搜索最优方案,该文将问题求解映射成青蛙在石头上觅食的过程,将任意资源调度方案 $Z(p)=(z_1^p,z_2^p,\cdots,z_n^p)$ 看作青蛙的位置,其中 w 维向量 $z_n^p=(x_{1n}^p,x_{2n}^p,\cdots,x_{wn}^p)^{\mathrm{T}}$ 是物理服务器 n 上部署功能组件的情况。

3.2 适应度函数设计

为指导算法迭代搜索,以公式(6)为适应度函数综合评价方案的优劣。

$$\mathrm{Fitness}(Z(p))=\beta\left(\sqrt[nr]{\prod_{a=1}^{r}\frac{\sum_{k=1}^{n}\sum_{i=1}^{w}(x_{ik}^p\times q_a^i)}{\sum_{k=1}^{n}(y_k^p\times t_a^k)}}\right)+(1-\beta)\sqrt[|G^p|r]{\prod_{k=1}^{n}F(y_k^p)} \tag{6}$$

其中,调度方案 $Z(p)$ 的适应度函数由 $\mathrm{Fitness}(Z(p))$ 计算,取值越高越好;β 为目标一的重要程度,日常和汛期会有不同取值;$|G^p|$ 为调度方案 $Z(p)$ 中重要核心功能组件占有物理服务器的数量。

3.3 初始调度方案生成

为迭代搜索最优资源调度方案,算法首先生成若干初始方案作为迭代搜索的基础。方案生成的主要思想是随机产生所有功能组件的部署顺序,优先采用贪婪策略部署重要核心功能组件,然后按照顺序部署其余功能组件。具体步骤如下:

步骤 1:生成重要核心功能组件的部署方案。按照重要核心功能组件的生成顺序,根据贪婪策略依次生成部署方案。策略以问题目标二最大化为指标,依次判断当前物理服务器是否能够满足约束条件,若满足则选择取值最大的物理服务器为部署方案。若存在重要核心功能组件在部署时无法满足约束,则重新生成部署顺序,依照步骤 1 生成可行方案。

步骤 2:部署其余功能组件生成方案。根据生成的部署顺序,对其余功能组件,以适应度函数值最大为目标,采用贪婪策略依次生成部署位置。生成过程中无法满足约束,则重新生成部署顺序,执行步骤 1 和步骤 2,完成初始方案生成。

3.4 蛙跳算子定义

为迭代搜索方案,重新定义蛙跳算子如下:

定义 1 差异算子⊙:用于分析两个资源调度方案在相应物理服务器上的差异。局部最优资源调度方案 $Z(pb)$ 与局部最差资源调度方案 $Z(pw)$ 在物理服务器上的差异可由公式(7)计算获得。

$$Z(pb)\odot Z(pw)=\{z_1^{pb}\odot z_1^{pw},z_2^{pb}\odot z_2^{pw},\cdots,z_n^{pb}\odot z_n^{pw}\} \tag{7}$$

$$z_n^{pb}\odot z_n^{pw}=\begin{cases}1,\sum_{i=1}^{w}|x_{in}^{pb}-x_{in}^{pw}|>0\\0,\sum_{i=1}^{w}|x_{in}^{pb}-x_{in}^{pw}|=0\end{cases} \tag{8}$$

定义 2 随机选择算子 rand( ):用于从资源调度方案间存在差异的物理服务器集合中随机选择若干物理服务器的操作。被选择的物理服务器由 n 维向量 D_i 表示,满足 $|D_i|\in[0,n]$。

定义 3 重新部署算子 ⊕:重新部署资源调度方案 $Z(pw)$ 在 D_i 上功能组件。具体步骤包括:清除被选择物理服务器上的部署方案,生成待部署功能组件集合。采用初始调度方案的生成策略,分别对集合中的重要核心功能组件和其他功能组件进行重新部署。

基于以上定义,局部最优资源调度方案 $Z(pb)$ 对局部最差资源调度方案更新,生成新方案 $Z(\mathrm{new})$ 的过程可由公式(9)和公式(10)表示。

$$D_i=\mathrm{rand}()*Z(pb)\odot Z(pw) \tag{9}$$

$$Z(\mathrm{new})=Z(pw)\oplus D_i \tag{10}$$

3.5 局部最优交叉操作

为快速搜索最优资源调度方案,算法将加强族群之间的交流,提升全局搜索能力。采用文献[17]的交叉框架,设计局部最优交叉策略,提升算法的搜索能力。具体如下:

步骤 1:构建局部最优交叉操作集合。选择每个种群中的最优方案,根据其适应度取值从大到小排序,选择顺序相邻的方案,两两配对构成局部最优交叉操作集合。

步骤 2:随机选择交叉点,调换交叉点前的功能组件。针对任意配对的局部最优调度方案,从 n 台物理服务器里选择除第一台以外的任意机器作为交叉点,将二者在交叉点前部署功能组件进行对调,生成两个新部署方案。

步骤 3:调整新部署方案,生成新资源调度方案。判断新部署方案是否存在重复和未部署功能组件,若存在,则选择新部署方案中重复功能组件对应的物理服务器,清除被选择物理服务器上的部署方案,同时加入未部署功能组件,生成待部署功能组件集合。利用重新部署算子生成新资源调度方案。

步骤 4:对比新资源调度方案与原局部最优调度方案的适应度取值,选择值大的方案参与下次迭代搜索。

3.6 最优青蛙变异操作

为进一步提升算法的迭代搜索能力,每次迭代都将对种群中的当前全局最优青蛙实施变异操作。即,随机选择当前全局最优资源调度方案中承载功能组件

的一台物理服务器,清除被选择物理服务器上的部署方案,生成待部署功能组件集合,利用重新部署算子对其进行重新部署,生成新资源调度方案。对比新资源调度方案与原全局最优调度方案的适应度取值,选择值大的方案作为新的全局最优方案。

3.7　算法的执行过程

基于改进蛙跳算法的气象云平台资源调度方法执行过程如下:

步骤 1:算法初始化设置。输入气象云平台的物理服务器情况(物理服务器的总数 n 、气象云平台的资源情况 $\{T^1,\cdots,T^n\}$)、功能组件的情况(功能组件的总数 w 、功能组件的等级 $\{L^1,\cdots,L^w\}$ 、功能组件的资源需求 $\{Q^1,\cdots,Q^w\}$ 、重要核心功能组件的标识 G)、服务器稳定性约束(不同等级功能组件对资源的预留要求 $\{S^1,\cdots,S^l\}$)、气象业务部署约束(功能组件之间的依赖关系 D ,功能组件与服务器之间冲突关系 C)、资源利用率的重要程度 β 等信息。设置方法的种群规模 gp、最大迭代次数 IT、子群的规模 tp。

步骤 2:生成初始资源调度方案集合,划分若干青蛙子群。根据种群规模设置,利用 3.3 节的方法生成初始资源调度方案集合 $\{Z(1),Z(2),\cdots,Z(\mathrm{gp})\}$ 。利用公式(6)计算每个方案的适应度,根据子群规模设置,结合适应度取值大小依次将生成方案划分到相应子群中,生成 m 个子群 $\{\mathrm{SP}^1,\cdots,\mathrm{SP}^m\}$ 。将子群中适应度最高和最低的方案设置为局部最优资源调度方案和局部最差调度方案。

步骤 3:更新青蛙子群内部的最差资源调度方案。首先利用差异算子分析局部优资源调度方案与局部最差调度方案的差异,使用随机选择算子获取局部最差调度方案需要调整的物理服务器,利用重新部署算子生成新的部署方案,判断其与原局部最差调度方案的适应度取值大小,若大于则利用新部署方案更新局部最差调度方案。

步骤 4:更新青蛙子群的局部最优资源调度方案。首先依据适应度对局部最优方案 $Z(pb^1),\cdots,Z(pw^m)$ 逆序排列生成 $Z(pb_1),\cdots,Z(pw_m)$ 和二者之间的映射关系 $\mathrm{RE}=\{\mathrm{re}_1,\mathrm{re}_2,\cdots,\mathrm{re}_m\}$ 。然后利用3.5 节中的局部最优交叉操作更新各个子群中的局部最优资源调度方案。

步骤 5:更新青蛙种群的全局最优资源调度方案。利用 3.6 节最优青蛙变异操作更新全局最优青蛙的资源调度方案 $Z(\mathrm{gbest})$ 。

步骤 6:迭代搜索最优资源调度方案,输出最终结果。判断算法是否小于最大迭代次数,若是则跳转至步骤 2 继续搜索,否则输出全局最优资源调度方案作为最终结果。具体如下所示:

算法:基于改进蛙跳算法的气象云平台资源调度方法。

输入: n,w,C,D , $\{T^1,\cdots,T^n\}$, $\{L^1,\cdots,L^w\}$, $\{S^1,\cdots,S^l\}$, $\{Q^1,\cdots,Q^w\}$, G , β ,gp,IT,tp

1. 初始化方案划分子群

$\{\mathrm{Z}(1),\mathrm{Z}(2),\cdots,\mathrm{Z}(\mathrm{gp})\}=\mathrm{Initial}()$　#根据输入生成初始调度方

$\{\mathrm{SP}^1,\cdots,\mathrm{SP}^m\}=\mathrm{Group}(\mathrm{Z}(1),\mathrm{Z}(2),\cdots,\mathrm{Z}(\mathrm{gp}),\mathrm{tp})$#划分成 m 个子群

2. 迭代搜索方案

For $i=1:\mathrm{IT}$　do

For $j=1:m$ do #更新子群最差调度方案

$Z(pb^j),Z(pw^j)\in \mathrm{SP}^j$ #子群的最优最差方案

$D_i=\mathrm{rand}()*Z(pb^j)\odot Z(pw^j)$

$Z(\mathrm{new})=Z(pw^j)\oplus D_i$

If $\mathrm{Fitness}(Z(pw^j))<\mathrm{Fitness}(Z(\mathrm{new}))$ Then

$Z(pw^j)=Z(\mathrm{new})$

End for

#依据适应度逆序排列局部最优方案

$\{[Z(pb_1),\ldots,Z(pb_m)],\mathrm{RE}\}=\mathrm{SortByFitness}(Z(pb^1),\cdots,Z(pb^m))$

For $k=1:\lfloor m/2\rfloor$ do　#进行局部最优交叉操作

$\{Z(\mathrm{new}_{2k}),Z(\mathrm{new}_{2k-1})\}=\mathrm{Cross}(Z(pb_{2k}),Z(pb_{2k-1}))$

If $\mathrm{Fitness}(Z(pb_{2k}))<\mathrm{Fitness}(Z(\mathrm{new}_{2k}))$ Then

$Z(pb_{2k})=Z(\mathrm{new}_{2k})$

If $\mathrm{Fitness}(Z(pb_{2k-1}))<\mathrm{Fitness}(Z(\mathrm{new}_{2k-1}))$ Then

$Z(pb_{2k})=Z(\mathrm{new}_{2k})$

End for

#更新子群的局部最优方案

$Z(pb^1),\cdots,Z(pb^m)=\mathrm{Update}([Z(pb_1),\cdots,Z(pb_m)],\mathrm{RE})$

$Z(\mathrm{gbest})=\mathrm{FindGBest}(Z(pb^1),\cdots,Z(pw^m))$ #找到种群最优方案

$Z(\mathrm{gnew})=\mathrm{Mutation}(Z(\mathrm{gbest}))$#全局最优变异操作

If $\mathrm{Fitness}(Z(\mathrm{gbest}))<\mathrm{Fitness}(Z(\mathrm{gnew}))$ Then

$Z(\mathrm{gbest})=Z(\mathrm{gnew})$

End for

输出:种群最优调度方案 $Z(\mathrm{gbest})$

4　实验仿真

实验以气象云平台的资源利用率和对重要核心业务的支撑情况为评价指标,综合分析方法的效果。气象云平台的资源利用率包含 CPU、内存、I/O、存储等指标,其对重要核心功能组件的支撑情况通过预留资源的多少来衡量。总体调度效果可由公式(6)来计算。考虑气象业务具有周期性特点,仿真将设计汛期和日常两种状态进行调度,其中气象云平台在日常将侧重于提升资源利用水平,汛期将侧重于对重要核心功能组件的支撑保障,方法评价的优劣主要体现在气象云平台汛期和日常的业务目标上。重要核心功能组件的资源需求可根据数据量和计算复杂度采用类比法预

估。在日常天气过程较少,气象应用对基础设施资源的需求相对稳定,虽然,不同功能组件在一天中的负载略微起伏,但是由于模型为每台物理服务器都预留了一定的基础资源,算法基本可以保证气象云平台的资源不会超额分配,从而应对负载有一些变化的场景。在汛期由于重大天气过程和气象保障需求的增多,重要核心功能组件将需要更多资源以应对业务的需要。为此,气象云平台需要针对其预留更多的资源以确保平台尽全力支撑重要核心功能组件的需求。

为验证方法的有效性,实验对比贪婪方法进行分析。贪婪方法首先利用 3.3 节的方法生成初始资源调度方案,然后再选择最优调度方案作为问题的解。该文在一台物理机中搭建仿真环境模拟资源调度平台,它能根据气象应用的资源需求和气象云平台的资源情况生成最终的调度方案。在仿真环境中,气象云平台的资源情况参考新疆气象云平台的能力生成,气象功能组件对气象云平台的资源需求及约束条件将结合气象应用在日常和汛期的情况随机生成,仿真主要参数如表 2 所示。

表 2 实验参数设置

功能组件个数	调度场景	功能组件的等级
60	{汛期,日常}	{重要核心,关键,一般}
70	{汛期,日常}	{重要核心,关键,一般}

基于以上实验数据,基于改进蛙跳算法的气象云平台资源调度方法和贪婪方法在汛期和日常,根据不同数量的功能组件的需求,对气象云中资源调度获得的效果如图 2 ~ 图 6 所示。其中,图 2 是两种方法在汛期和日常,根据不同数量的功能组件的需求获得的调度效果,图 3 和图 4 展示了基于改进蛙跳算法的气象云平台资源调度方法对气象云平台资源调度后的资源利用率情况。图 5 和图 6 展示了基于改进蛙跳算法的气象云平台资源调度方法对重要核心功能组件的平均资源预留比例。

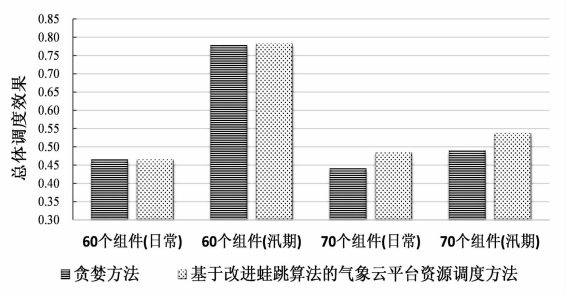

图 2 两种方法在汛期和日常对气象云平台资源调度的综合效果对比

从图 2 可以看出,基于改进蛙跳算法的气象云平台资源调度方法在汛期和日常,针对不同功能组件的需求,与贪婪方法相比,均能取得较好的总体调度效果。由于贪婪方法初始时采用随机方式生成部署顺序,因此有一定概率获得较好的调度方案,但方法并不稳定。该方法在 60 个功能组件时取得了较为良好的效果,但在 70 个功能组件时效果并不理想。基于改进蛙跳算法的气象云平台资源调度方法是在贪婪方法的基础上不断迭代改进,因此其效果总体优于贪婪方法。

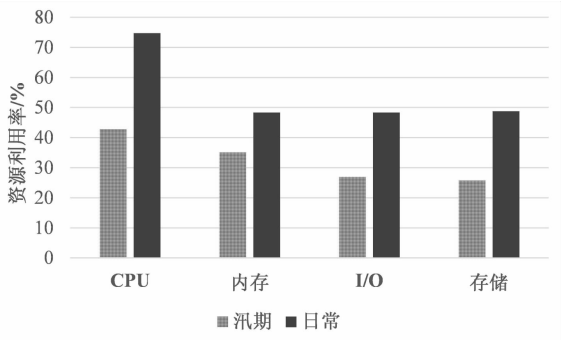

图 3 文中方法在汛期和日常对气象云平台资源调度后的资源利用率(60 个功能组件)

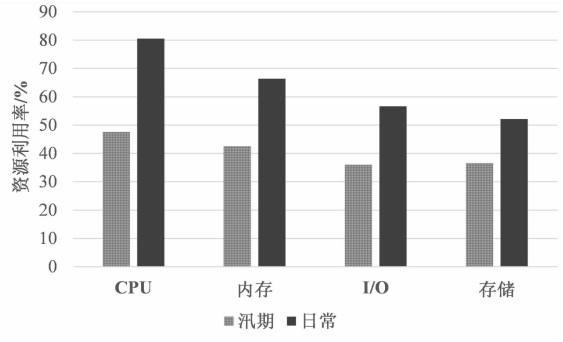

图 4 文中方法在汛期和日常对气象云平台资源调度后的资源利用率(70 个功能组件)

从图 3 和图 4 可以看出,文中方法能够根据汛期和日常要求,动态调整气象云平台的资源以支撑不同数量功能组件的需求。在日常本方法能使气象云平台保持较高资源利用水平。

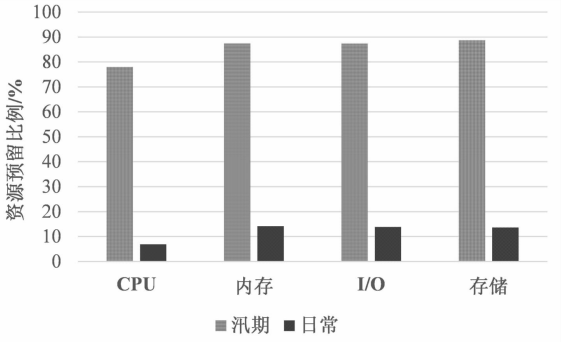

图 5 文中方法在汛期和日常对重要核心功能组件平均预留的资源比例(60 个功能组件)

从图 5 和图 6 可以看出,文中方法能够根据汛期和日常要求,动态调整气象云平台对重要核心功能组件的平均预留资源比例,在汛期本方法能使气象云平台预留较多的基础资源,保证重要核心业务安全稳定

运行。

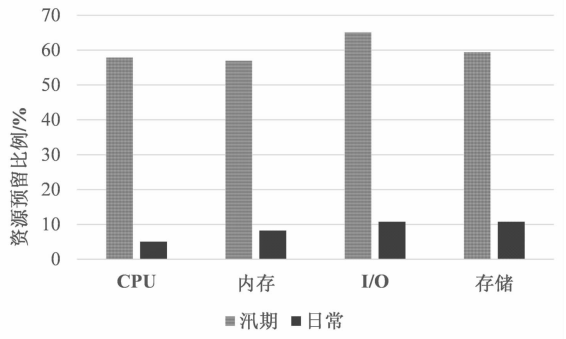

图 6　文中方法在汛期和日常对重要核心功能组件平均预留的资源比例(70 个功能组件)

综上所示,基于改进蛙跳算法的气象云平台资源调度方法针对不同数量的功能组件需求,总能够获得较好的总体调度效果,与贪婪方法相比该方法的搜索能力更强。

5　结束语

气象云平台是集约支撑气象业务发展基础资源平台,合理调度气象云中的基础资源是气象业务稳定运行的关键。该文根据气象业务在日常和汛期对基础设施资源需求的特点,设计了两个目标函数来分别评价气象云平台的资源利用水平和其对重要核心业务的支撑保障,并在约束中考虑了功能组件与物理服务器的冲突关系等因素,将问题建模成一个多目标优化问题。然后,通过适应度函数来综合评价模型目标,设计一种改进蛙跳算法的气象云资源调度方法。方法通过重新定义蛙跳算子,利用局部最优交叉操作和最优青蛙变异策略来迭代搜索问题的资源调度方案。最后,通过实验仿真验证了方法的效果。

参考文献:

[1] 王　彬,韩同欣,李　楠. 气象私有云环境下存储架构设计与性能分析[J]. 计算机技术与发展,2017,27(5):20-24.

[2] 徐建鹏,李　欣,赵晓凡. 一种云存储环境下的资源调度改进算法[J]. 计算机应用研究,2019,36(7):2015-2019.

[3] 师雪霖,徐　恪. 云虚拟机资源分配的效用最大化模型[J]. 计算机学报,2013,36(2):252-262.

[4] 史宝鹏,段　迅,孔广黔,等. 医疗云平台资源调度策略研究[J]. 计算机工程,2017,43(8):44-48.

[5] 刘　愉,赵志文,李小兰,等. 云计算环境中优化遗传算法的资源调度策略[J]. 北京师范大学学报:自然科学版,2012,48(4):378-384.

[6] 童　钊,邓小妹,陈洪剑,等. 云环境下基于强化学习的多目标任务调度算法[J]. 小型微型计算机系统,2020,41(2):285-290.

[7] 李　超,戴炳荣,旷志光,等. 云计算环境下基于改进遗传算法的多维约束任务调度研究[J]. 小型微型计算机系统,2017,38(9):1945-1949.

[8] 马小晋,饶国宾,许华虎. 云计算中任务调度研究的调查[J]. 计算机科学,2019,46(3):1-8.

[9] 张丽梅. 基于负载均衡的云资源调度策略研究[D]. 银川:宁夏大学,2014.

[10] REN X, ANANTHANARAYANAN G, WIERMAN A, et al. Hopper:decentralized speculation-aware cluster scheduling at scale[C]//The ACM special interest group on data communication. London:ACM,2015:379-392

[11] CUI L, HAO Z, PENG Y, et al. Piccolo:a fast and efficient rollback system for virtual machine clusters[J]. IEEE Transactions on Parallel and Distributed Systems, 2017, 28(8):2328-2341.

[12] 王煜炜,刘　敏,房秉毅,等. 面向网络性能优化的虚拟计算资源调度机制研究[J]. 通信学报,2016,37(8):105-118.

[13] 陈黄科,祝江汉,朱晓敏,等. 云计算中资源延迟感知的实时任务调度方法[J]. 计算机研究与发展,2017,54(2):446-456.

[14] AL-MOALMI A, LUO J, SALAH A, et al. A whale optimization system for energy-efficient container placement in data centers[J]. Expert Systems with Applications, 2020, 164:113719.

[15] 韦传讲,庄　毅. 面向移动云计算的自适应虚拟机调度算法[J]. 小型微型计算机系统,2021,42(1):96-104.

[16] RAMPERSAUD S, GROSU D. Sharing-aware online virtual machine packing in heterogeneous resource clouds[J]. IEEE Transactions on Parallel and Distributed Systems, 2017, 28(7):2046-2059.

[17] 郭　静. 面向私有云的业务迁移部署方法研究[J]. 中国电子科学院学报,2016,11(2):191-198.

[18] 苏命峰,王国军,李仁发. 基于利益相关视角的多维 QoS 云资源调度方法[J]. 通信学报,2019,40(6):102-115.

[19] 谷南南,姚佩阳,焦志强. 云计算环境下利用改进遗传算法结合二次编码的大规模资源调度方法[J]. 计算机应用研究,2020,37(8):2390-2394.

[20] MANN Z A. Resource optimization across the cloud stack[J]. IEEE Transactions on Parallel and Distributed Systems,2018,29(1):169-182.

[21] SONKLIN C, TANG M, TIAN Y C. A decrease-and-conquer genetic algorithm for energy efficient virtual machine placement in data centers[C]//IEEE international conference on industrial informatics. Emden:IEEE,2017:135-140.