DOI: 10.11992/tis.201710004
网络出版地址: http://kns.cnki.net/kcms/detail/23.1538.TP.20180413.1722.008.html

负载敏感的云任务三支聚类评分调度研究

吴俊伟，姜春茂
（哈尔滨师范大学 计算机科学技术与信息工程学院，黑龙江 哈尔滨 150025）

摘　要：在云计算商业化的服务模式中，追求服务质量、负载均衡与经济原则的多目标优化调度。针对集群资源使用率偏低的现象，提出了三支聚类评分 (three-way clustering weight, TWCW) 算法，首先分析云任务的多样化需求与资源的动态特性，采用三支聚类算法对任务集合聚类划分，然后结合任务属性对类簇对象进行评分调度。基于 Cloudsim 实验模拟表明：相比于 k-means 与 FCM 聚类调度，三支聚类评分算法 (TWCW) 在任务平均响应时间与资源利用率等方面均有显著提升。

关键词：云计算；优化调度；多样化需求；动态资源；三支聚类；评分调度；任务响应时间；资源使用率
中图分类号：TP311.13　　**文献标志码**：A　　**文章编号**：1673−4785(2019)02−0316−07

中文引用格式：吴俊伟，姜春茂. 负载敏感的云任务三支聚类评分调度研究[J]. 智能系统学报, 2019, 14(2): 316–322.
英文引用格式：WU Junwei, JIANG Chunmao. Load-aware score scheduling of three-way clustering for cloud task[J]. CAAI transactions on intelligent systems, 2019, 14(2): 316–322.

Load-aware score scheduling of three-way clustering for cloud task

WU Junwei, JIANG Chunmao
(School of Computer Science Technology and Information Engineering, Harbin Normal University, Harbin 150025, China)

Abstract: A commercialized model is established for multi-objective optimization scheduling of service quality, balanced load, and economic principles in cloud computing. A three-way clustering weight (TWCW) algorithm is proposed to solve the problem of the low utilization rate of cluster resources. First, the diversified requirements of cloud tasks and the dynamic characteristics of resources are analyzed to cluster and divide the task set by the TWCW algorithm and then score scheduling by combination with task attributes. Simulation results based on Cloudsim show that compared with k-means and FCM clustering scheduling, the TWCW algorithm has significant improvements in the average task response time and resource utilization rate.

Keywords: cloud computing; optimal scheduling; diversified requirement; dynamic resource; three-way clustering; scoring scheduling; response time of task; resource utilization rate

在云计算提供强大服务的背后，存在集群资源使用率偏低的现象：Bohrer[1]指出集群节点的使用率一般仅为 11%~50%，大量资源处于闲置状态，资源浪费严重。Barrsos 等[2]通过统计分析近 5 000 台主机近半年的运行状况，节点的使用率仅有 10%~50%。因此，如何准确有效地分配云资源，提高集群资源利用率是云计算的一个重要研究内容。

在云计算的研究中，通过对 Google cluster trace 分析发现[3-4]，云任务可分为即时响应的应用与批处理任务，其服务特征、资源需求具有多样性。多数研究通过分析任务作业时长，资源使用特性等，对任务进行聚类[5-10]，以此描述任务的异构性特征，并根据聚类后各类簇的特性设置调度函数，其实验结果表明聚类调度对任务响应时间与集群资源使用有显著改善。但任务聚类通常并不完全是非此即彼的类别划分，也可能出现中间态模糊集。针对此类问题，三支决策聚类通过定义边缘域概念拓展了传统的二支聚类算法，并根

收稿日期：2017−10−10.　**网络出版日期**：2018−04−16.
基金项目：中国博士后面上基金项目 (2014M561330).
通信作者：吴俊伟. E-mail: 1344845860@qq.com.

据对象属性对兼具多标签对象作出进一步的划分, 其中, 三支决策理论源于 Yao 对概率粗糙集和决策粗糙集 3 个域的合理解释[11-13], 是一种更为一般, 有效的决策方法。本文将其引入云计算, 为任务调度问题提供新的解决方案。

1　相关工作

作为一种有效的问题解决手段, 三支决策的基本思想是通过一对阈值(α,β)将一个全集U划分成 3 个独立的部分, 然后针对各个区域设置相应的策略。其特征是使用三支方法进行问题解决和信息处理。三支决策基于启发式方法, 将复杂问题利用分治策略转化为简单问题, 其定义为

$$c_{(\alpha,\beta)}:\tau:U\to\{S_1,S_2,S_3\} \tag{1}$$

式中: 基于条件集C, 三支决策通过映射τ将实体集U分成 3 个两两互不相交的S_1、S_2、S_3,然后根据 3 个区域的特点, 有针对性的设计策略和动作, 以期达到某种收益的最大化。关于三支决策的理论, 模型与应用研究已经取得了较大进展, 如垃圾邮件过滤研究[14]、代价敏感的三支决策模型研究[15]、三支决策和博弈论[16]、多粒度三支决策[17]、序列三支决策[18]、动态三支决策[19]、三支决策与逻辑[20]等。

作为三支决策在聚类算法方面的有效应用, 三支聚类[21-22]拓展了对象与集合的隶属关系。给定对象集合$U=\{x_1,x_2,\cdots,x_n\}$, n表示对象数。以二支聚类的思想判定, 对象x_i, 要么属于集合cs_i, 要么不属于集合cs_i, 其聚类的结果为$CS=\{cs_1,cs_2,\cdots cs_k\}$, k为类别数。而以三支的角度观察, 对象x_i与集合cs_i的所属关系有: x_i属于cs_i, x_i可能属于cs_i, x_i不属于cs_i, 因此集合cs_i被划分为互不相交的L域、M域和R域, 其中L域表示该类簇的核心对象集合, M域为边缘对象集合。其聚类结果为$CS=\{(L(cs_1),M(cs_1)),(L(cs_2),M(cs_2)),\cdots,(L(cs_k),M(cs_k))\}$。三支聚类算法根据簇间离散度与簇内聚合度, 对类中对象之间的紧密程度作进一步划分。

传统的二支聚类算法在云计算中被广泛应用。文献[5]使用 k-means 算法将任务聚类为计算、存储与网络类型, 引入权重因子调整属性间优先级, 根据任务类型调度任务。刘家志[6]将 FCM 算法引入到任务调度中, 根据 CPU、内存、IO 与带宽等属性为任务建模并聚类, 根据类簇特征为任务设置调度函数。文献[7]根据集群资源的多维属性定义资源可见度, 通过静态阈值划分可见度等级, 在 PSO 调度中以该等级与任务执行时间作为适应性函数, 提高集群负载均衡度。张以利等[8]根据作业长短及重要性聚类任务, 构建 LPM 模型, 追求最大化的任务完成数, 文献[9]采用层级聚类对云任务进行预处理, 以最小化任务完成时间为目标函数调度任务, 兼顾资源负载与系统吞吐量, 在满足服务质量的同时, 缩短任务完成时间。高正九等[10]提出了一种基于任务分类的延迟调度算法, 根据任务长度聚类, 并依据该类别调整任务等待时间阈值。相比于 DS、FIFO 算法, 该算法有效缩短了任务响应时间。

三支决策理论体系的引入, 将对任务调度提供新思路。本文提出的三支聚类评分 (three-way clustering weight, TWCW) 算法, 对任务进行三支聚类划分, 追求最大化资源使用, 根据聚类结果中核心态与模糊态任务偏好设置调度策略。

2　TWCW 算法

2.1　任务模型

在 2011 公布的 Google Cluster Trace 中, 作业由一个或多个任务构成, 与任务相关的属性有优先级、资源请求 (resource request) 与限制条件等[3]。从任务优先级的角度观察, Google 设置了任务的 12 个优先级, 并将其依次划分为高优先级 (9-11)、中优先级 (2-8) 与低优先级 (0-1) 三类。优先级越高, 意味着成功调度的几率更大, 拥有更高的服务质量。且各优先级作业, 其运行时长, 资源使用特征存在异构性。在资源请求的维度上, 任务请求的资源主要有 CPU 与内存空间, 通过对 CPU 与内存在请求率上的回归分析得出, 两者存在关联, 但关联性较弱 ($R\approx0.14$)。在任务的限制条件中, 通常是由用户为作业指定主机性能或是其他关联性任务, 限制条件数量与任务的延迟调度并不存在明显关联关系, 例如, 对于只有一个限制条件的任务集合, 与同时拥有 6 个限制条件的任务集合, 其平均调度延迟差值较小。

在 trace 数据中, 作业可分为面向用户的交互式作业与批处理作业, 它们对资源的请求存在多样化特性, 且主要体现为对计算、内存与带宽资源的请求。因此本文选择在资源请求的维度为任务建模, 以(id,mips,mem,bw)元组的形式描述任务的资源请求, 其中, id表示任务标识, mips表示任务请求的计算资源, 执行任务指令。mem表示任务请求的内存资源, 用于构建程序数据结构, bw表示请求的带宽资源, 用于访问任务链接的外部资源。

2.2　调度模型

调度器是集群的核心, 负责任务的调度与资源的合并、迁移, 其调度方式与效率将极大影响

集群的性能输出。本文建立的调度模型如图 1 所示，其主要模块和功能如下：

1) 调度器 (Scheduler)：聚类调度，为任务选择合适节点。

2) 计算节点 (Host)：物理机节点，承载虚机。

3) 虚机 (VM)：资源分配的基本单元，执行请求任务。

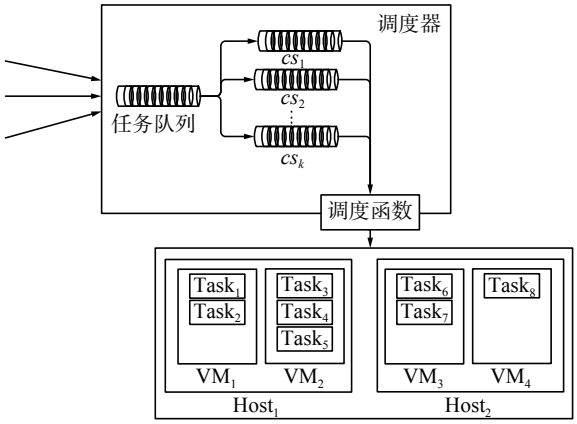

图 1　集群架构图

Fig. 1　Cluster architecture

如图 1 所示，用户任务 (Task) 进入任务队列 (Task queue)，调度器 (Scheduler) 首先提取 Task queue 中的任务集合，根据任务属性将其聚类划分为$\{cs_1, cs_2, \cdots, cs_k\}$，然后通过调度函数 (Strategies) 为各类簇任务选择目标主机 (Host)，由 Host 中的虚机 (VM) 执行该任务，将结果返回。

2.3　算法描述

给定任务集合$T = \{t_1, t_2, \cdots, t_n\}$，其中$t_i = \{\mathrm{id}, \mathrm{mips}_i, \mathrm{mem}_i, \mathrm{bw}_i\}$，分别表示任务 i 标志，请求的计算、内存、带宽资源。设有主机列表$H = \{h_1, h_2, \cdots, h_m\}$，其中 $h_i = \{(P_i, p_i), (M_i, m_i), (B_i, b_i)\}$，$P_i$、$M_i$、$B_i$ 分别表示 h_i 最大可分配的计算、内存与带宽资源，p_i、m_i、b_i 则分别表示 h_i 已分配的计算、内存、带宽资源。TWCW 算法由聚类，评分与调度 3 部分组成。

2.3.1　聚类

基于三支聚类算法，提出任务类簇的三支表示形式：

$$cs_i = (L(cs_i), M(cs_i)) \tag{2}$$

式中：$L(cs_i) \subseteq T$ 且 $M(cs_i) \subseteq T$。设 $R(cs_i) = T - L(cs_i) - M(cs_i)$，则 $L(cs_i)$、$M(cs_i)$、$R(cs_i)$ 构成了类簇 cs_i 的核心域、边缘域和琐碎域，它们满足如下性质：

$$T = L(cs_i) \cup M(cs_i) \cup R(cs_i) \tag{3}$$

$$\begin{aligned} L(cs_i) \cap M(cs_i) &= \varnothing \\ L(cs_i) \cap R(cs_i) &= \varnothing \\ M(cs_i) \cap R(cs_i) &= \varnothing \end{aligned} \tag{4}$$

若任务 $t \in L(cs_i)$，则 t 属于类簇 cs_i；若任务 $t \in R(cs_i)$，则 t 不属于类簇 cs_i；若任务 $t \in M(cs_i)$，则 t 可能属于类簇 cs_i。其聚类结果为 $= \{(L(cs_1), CSM(cs_1)), \cdots, (L(cs_k), M(cs_k))\}$。三支聚类算法包括寻找最佳聚类数与确定类簇域对象两个子步骤，其基本思想如下：

1) 寻找最佳聚类数，使用聚类算法 (如 k-means) 聚类任务集合 T，根据类簇间离散度与类簇内聚合度择优确定最佳聚类结果；

2) 确定类簇域对象，首先通过近邻域确定类簇 M 域对象，然后使用差值排序进一步划分类簇 L 域与 M 域对象，其具体内容如下：对于类簇 cs_k，任务 t_i、t_j 与近邻域 $\mathrm{Neig_q}(t_i)$，其中，类簇 cs_k 存在中心点 $\mathrm{centroid_k}$，$\mathrm{Neig_q}(t_i)$ 表示在欧式距离上离 t_i 最近的 q 个任务，当 $t_i \notin cs_h$，$t_j \in Neig_q(t_i)$ 时，若 $t_j \in cs_h$，则确定 t_i 为 $M(cs_h)$ 对象，然后根据类簇 cs_h 内任务集合 $t_1, t_2, \cdots, t_l$ 与中心点 $\mathrm{centroid_h}$ 的欧式距离差值排序，确定差值最大的任务对 t_{p-1} 和 t_p，将 $t_1, t_2, \cdots, t_{p-1}$ 划分到 $L(cs_h)$ 中，$t_p, t_{p+1}, \cdots, t_l$ 划分给 $M(cs_h)$。其算法描述如下：

输入　任务集合 T，近邻数 q。

输出　$CS = \{(L(cs_1), M(cs_1)), \cdots, (L(cs_k), M(cs_k))\}$

1) 初始化，$k = 2$；

2) 随机选取 k 个聚类中心 $v_1, v_2, \cdots, v_k$；

3) 对于每个任务 t_i，计算其到每个聚类中心 v_i 的距离，将其划分到距离最小的类中；

4) 更新聚类中心 $v = \{v'_1, v'_2, \cdots, v'_k\}$；

5) 如果聚类中心不发生变化至 6)；否则转至 3)；

6) 计算 $\mathrm{CVIN}(CS)$，如果 $k \leqslant \sqrt{N}$，那么 $k = k + 1$，转至 2)；否则转至 7)；

7) 考查任务 t_i 和类 cs，其中 $t_i \in cs, t_j \in \mathrm{Neig_q}(t)$。如果 $t_j \in cs$，那么把 t_i 划分到 $M(cs)$；

8) 对于类中剩余非 M 域中对象，根据差值排序法，找到第一个距离差值最大的对象对 t_{i-1} 和 t_i，把 t_i 及其后的对象划分到 $M(cs)$；

9) 算法结束，输出结果 $CS = \{(L(cs_1), M(cs_1)), \cdots, (L(cs_k), M(cs_k))\}$。

2.3.2　评分

在对任务集合 T 进行三支聚类后得到聚类结果 $CS = \{(L(cs_1), M(cs_1)), \cdots, (L(cs_k), M(cs_k))\}$，其中，类簇 cs_i 的类簇中心 $\mathrm{centroid_i} = \{\mathrm{mips}, \mathrm{mem}, \mathrm{bw}\}$。通过比较类簇中心 centroid 间属性大小确定类簇间属性比重与类簇内属性偏好，以评分矩阵的形式记录评分结果。

定义 1　类簇评分矩阵 $\boldsymbol{W}_{NxM}$，如式 (5) 所示。

$$\boldsymbol{W}_{N\times M} = \begin{bmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,M} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,M} \\ \vdots & \vdots & & \vdots \\ w_{N,1} & w_{N,2} & \cdots & w_{N,M} \end{bmatrix}, N = 3, M = k \tag{5}$$

式中: $w_{i,j}$表示类簇j第i维属性得分值; 矩阵的行向量$\boldsymbol{w}_i=[w_{i,1}\ w_{i,2}\ \cdots\ w_{i,M}]$表示各类簇中心在第$i$维资源上的评分值, 记录类簇间资源请求的比重; 矩阵的列向量$\boldsymbol{w}_j=[w_{1,j}\ w_{2,j}\ \cdots\ w_{N,j}]^{\mathrm{T}}$表示类簇$j$资源请求偏好。评分函数通过排序算法比较$\mathrm{centroid}_1$, $\mathrm{centroid}_2,\cdots,\mathrm{centroid}_k$间第$i$维属性, 确定行向量$\boldsymbol{w}=[w_{i,1}\ w_{i,2}\ \cdots\ w_{i,M}]$。假设评分矩阵为

$$\boldsymbol{W}_{3\times4}=\begin{bmatrix}4&3&2&1\\3&1&4&2\\2&4&1&3\end{bmatrix}$$

观察$\boldsymbol{W}$行向量可知, 对于请求的mips资源, $\mathrm{centroid}_1>\mathrm{centroid}_2>\mathrm{centroid}_3>\mathrm{centroid}_4$, 所以$w_{1,1}$的得分最高, $w_{1,2}$次之。接着观察其列向量, 以列向量$\boldsymbol{w}_1=[4\ 3\ 2]^{\mathrm{T}}$为例, 类簇$cs_1$中的任务请求 4 单位的mips, 3 单位的mem与 2 单位的bw。任务偏好为 4:3:2。

2.3.3　调度

在完成对任务的聚类、评分后, 根据聚类结果CS与评分矩阵$\boldsymbol{W}$, 分别对cs_i的L域、M域任务设置调度函数。TWCW 算法追求最大化的资源使用, 因此调度函数需充分利用闲置资源, 均衡集群负载。通过定义主机host_i资源剩余可分配空间表示其资源空闲度。

定义 2　主机host_i的第j维资源剩余可分配空间$\mathrm{space}_{i,j}$, 如式 (6) 所示。

$$\mathrm{space}_{i,j}=(\mathrm{allocable}_{i,j}-\mathrm{allocated}_{i,j})/\mathrm{allocable}_{i,j}\tag{6}$$

式中: $\mathrm{allocable}_{i,j}$表示host_i最大可分配的j维资源, $\mathrm{allocated}_{i,j}$表示host_i已分配的j维资源。

定义 3　主机host_i的资源剩余可分配空间, 如式 (7) 所示。

$$S_i=\sum_{j=1}^{N}\mu_j\cdot\mathrm{space}_{i,j},\ N=3\tag{7}$$

式中μ_j表示主机host_i的第j维资源的权重。

TWCW 算法可最大限度的利用空闲资源, 根据任务属性评估节点列表H的剩余可分配空间S_i, 选择资源可分配空间最大的主机作为目标节点, 如式 (8) 所示。

$$h_{\mathrm{target}}=\max(\sum_{j=1}^{N}\mu_j\mathrm{space}_{i,j}),\ N=3,i=1,2,\cdots,m\tag{8}$$

TWCW 算法可最大限度的利用空闲资源, 根据任务属性评估节点列表H的剩余可分配空间S_i, 选择资源可分配空间最大的主机作为目标节点, 如式 (8) 所示。

对于$L(C_i)$域的任务, 通过归一化评分矩阵$\boldsymbol{W}$列向量确定权重因子μ_j, 如式 (9) 所示。

$$\mu_j=\frac{w_{j,i}}{\sum_{j=1}^{N}w_{j,i}},N=3\tag{9}$$

对于$M(C_i)$域的任务, 根据任务请求的资源在集群资源中的所占比确定该任务请求的主资源, 将多维属性映射到单维空间, 从而确定权重因子μ, 其算法描述如下。

输入　任务t;

输出　权重因子μ。

1) 初始化权重因子μ;

2) 获取集群资源 resource, 初始化迭代变量 i=0;

3) 计算在第i维属性中, 任务t在资源 resource 中的比重 prop_i=t.attr$_i$/resource$_i$;

4) 如果$i<N$, $i=i+1$, 转至步骤 3)。否则转至 5);

5) 确定 prop_i 中比重最大的第l维属性, 将μ_l置为 1, 算法结束。

2.3.4　时间复杂度分析。

TWCW 算法包括了任务聚类、评分与调度, 其中:

1) 三支聚类时间复杂度分析。设任务集合的任务数为n, 聚类分支数为k, 近邻数为q, k-means 的迭代次数为I, 寻找最佳聚类数目的时间复杂度为$O(n^2I+n^2q)$, 确定分支域对象的时间复杂度为$O(n\log n+knq)$[21]。

2) 类簇评分时间复杂度分析。设聚类分支数为k, 确定类簇中心在单个属性的排序算法的时间复杂度为$O(k\log k)$。

3) 选择目标主机时间复杂度分析。设任务集合的任务数为n, 主机列表的节点数为m, 对于单个任务选择目标节点的时间复杂度为$O(m)$, 那么选择目标主机的时间复杂度为$O(nm)$。

3　实验及数据分析

3.1　实验环境与参数设置

本文采用 cloudsim3.0 进行实验。实验模拟 1 000 个节点, 主机均为四核的, 单核处理能力为 37 274 MIPS。主机 RAM 为 8 GB, 硬盘容量为 1 TB, 带宽 1 Gb/s。随机生成一系列虚拟请求构成的任务 (请求序列)。任务的相关参数如表 1 所示。

实验采用了两个指标来评估 TWCW 算法的性能: 平均响应时间和系统资源利用率。其中, 平均响应时间包括等待时间和处理时间, 系统资源利用率为主机集合的平均使用率。实验同时实现了 k-means[5]和 FCM[6]聚类调度算法, 观察它们在上述性能方面的表现。

表 1　任务参数

Table 1　Task parameters setup

参数	数值
长度 (CPU)/MIPS	[400, 1 000]
文件大小/MB	[200, 1 000]
内存/MB	[20, 60]

3.2　任务平均响应时间比较实验

评估了 TWCW 聚类调度在任务响应时间的表现。实验分别实现了 TWCW、k-means 和 FCM 算法来验证聚类方式的改变对任务响应时间的影响，实验结果分别如图 2 所示。

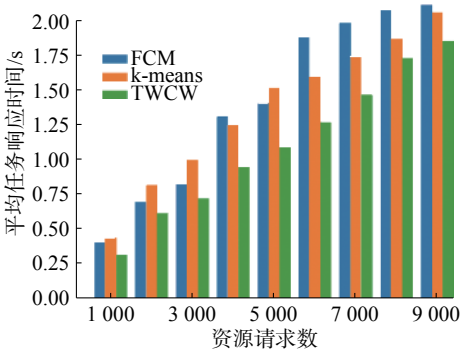

图 2　资源请求增加时响应时间对比

Fig. 2　The average response time comparison when requests are increased

由图 2 可以看出 TWCW 算法比 k-means 或 FCM 在任务响应时间上有一定下降。说明 TWCW 在对任务更细粒度聚类划分后，为任务选择偏好属性更充足的节点，使其能够更快地完成计算操作返回结果从而减少任务运行时间。经统计 TWCW 算法任务的平均响应时间，相比于 FCM 与 k-means 算法减少了约 7%。

3.3　集群资源利用率比较实验

验证了 TWCW 算法在资源利用率方面的优化效果。实验分别评估了 TWCW、k-means 和 FCM 算法在集群资源利用率方面的表现。实验结果分别如图 3~4 所示。

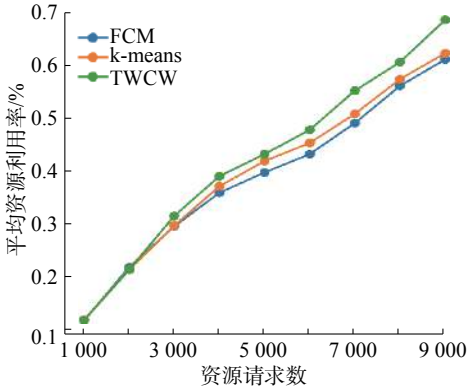

图 3　资源请求增加时平均利用率对比

Fig. 3　The resource average utilization comparison when requests are increased

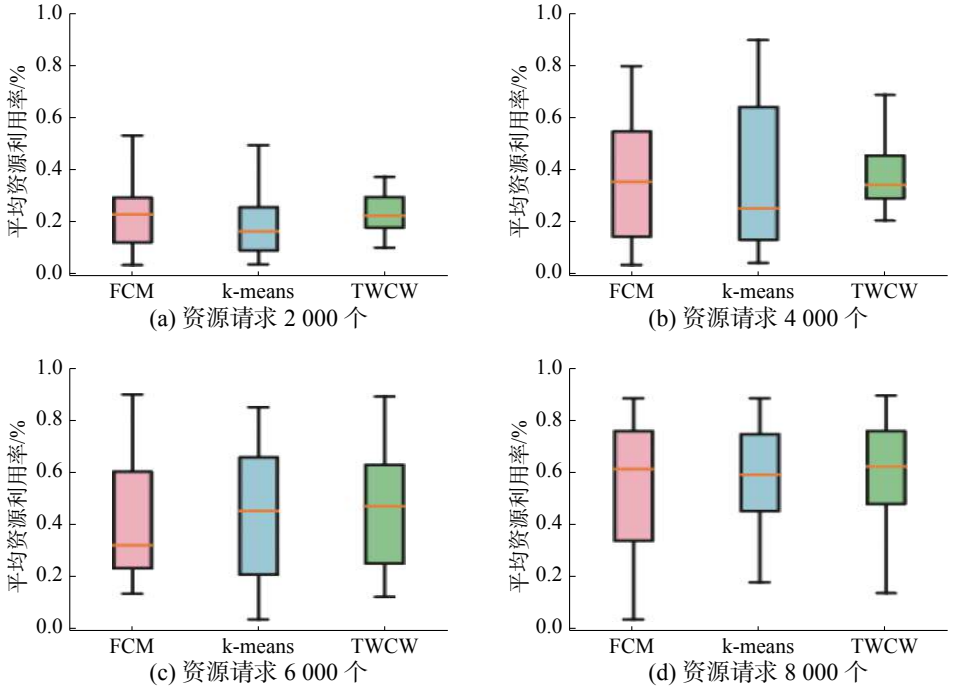

图 4　资源请求增加时资源使用情况对比

Fig. 4　The resource usage comparison when requests are increased

由图 3 可知，集群资源使用率随着负载的增加而增加，可以看出，在资源请求为 1 000~2 000 个时 3 种算法的差别不大，而在请求幅度继续增大时，TWCW 算法的优化效果较为明显，相对其他算法最高有近 11.3% 的改善。

图 4 中四分位区间内的实线表示资源利用率的中位数。可以观察到，TWCW 算法四分位区间相对集中且利用率中位数均高于其他算法，说明相比于 k-means或 FCM 需提前设置聚类分支，TWCW 聚类类簇的三支划分与域对象的分类调度能细粒度地将任务匹配给合适资源，避免了节点的过量负载与资源闲置，从而能更有效地平滑集群负载，提高资源使用率。

4 结束语

本文在云任务调度中, 引入三支聚类评分算法, 以提高资源利用率为目标, 利用任务的多样化特性, 对任务集合进行三支聚类, 通过类簇评分调度, 实现资源的合理分配。并在试验中实现并比较了 k-means 和 FCM 聚类调度算法, 考察它们在上述性能方面的表现, 实验结果表明: 对任务集合细粒度的类簇划分与对象的针对性调度, 有利于集群资源利用率与负载均衡。

在未来的工作中, 将对任务进行更细化的分类建模, 改进任务与资源的匹配模型, 且基于节能角度的考虑, 对资源负载进行动态迁移, 确定虚机迁移阈值的在线学习方案, 设计评价函数动态调整阈值区间等。

参考文献:

[1] BOHRER P, ELNOZAHY E N, KELLER T, et al. The case for power management in web servers[M]//GRAYBILL R, MELHEM R. Power Aware Computing. Boston, MA, USA: Springer, 2002: 261–289.

[2] BARROSO L A, HÖLZLE U. The case for energy-proportional computing[J]. Computer, 2007, 40(12): 33–37.

[3] REISS C, TUMANOV A, GANGER G R, et al. Heterogeneity and dynamicity of clouds at scale: Google trace analysis[C]//Proceedings of the third ACM Symposium on Cloud Computing. New York, NY, USA, 2012: 1–13.

[4] LIU Zitao, CHO S. Characterizing machines and workloads on a Google cluster[C]//Proceedings of 2012 41st International Conference on Parallel Processing Workshops. Pittsburgh, PA, USA, 2012: 397–403.

[5] 左利云. 云计算中基于任务特性和资源约束的调度方法研究[D]. 广州: 华南理工大学, 2016: 1–139.
ZUO Liyun. The scheduling methods based on the task features and resource constraints in cloud computing[D]. Guangzhou, China: South China University of Technology, 2016: 1–139.

[6] 刘家志. 模糊 C-均值算法在任务调度问题上的应用[C]//第十届中国通信学会学术年会论文集. 沈阳, 中国, 2014: 310–313.
LIU Jiazhi. Application of fuzzy c-means algorithm in task scheduling problem[C]//A Collection of Academic Annual Meetings of the Tenth China Communications Society. Shenyang, China, 2014: 310-313.

[7] 封良良, 夏晓燕, 贾振红, 等. 实验基于资源预先分类的云计算任务调度算法[J]. 计算机仿真, 2013, 30(10): 363–367, 410.
FENG Liangliang, XIA Xiaoyan, JIA Zhenhong, et al. Task scheduling algorithm based on improved particle swarm optimization algorithm in cloud computing environment[J]. Computer simulation, 2013, 30(10): 363–367, 410.

[8] 张以利, 杨万扣. 云环境下基于任务分类和 LPM 优化模型的调度算法[J]. 微型电脑应用, 2013, 29(10): 5–8.
ZHANG Yili, YANG Wankou. Scheduling algorithm based on task classifying and linear pogramming model in a cloud environment[J]. Microcomputer application, 2013, 29(10): 5–8.

[9] 陈晶晶. 云环境下基于非均匀粒度分类的任务调度算法研究[D].南京: 南京邮电大学, 2015: 1–59.
CHEN Jingjing. Research on task scheduling algorithm based on non-uniform granularity classification in cloud environment[D]. Nanjing, China: Nanjing University of Posts and Telecommunications, 2015: 1–59.

[10] 高正九, 郑烃, 辛波, 等. 基于任务分类的延迟调度算法[J]. 计算机系统应用, 2014, 23(9): 139–143.
GAO Zhengjiu, ZHENG Quan, XIN Bo, et al. Delay scheduling algorithm based on task classification[J]. Computer systemsand & applications, 2014, 23(9): 139–143.

[11] YAO Yiyu. The superiority of three-way decisions in probabilistic rough set models[J]. Information sciences, 2011, 181(6): 1080–1096.

[12] YAO Yiyu. Three-way decisions with probabilistic rough sets[J]. Information sciences, 2010, 180(3): 341–353.

[13] YAO Yiyu. Three-way decision: an interpretation of rules in rough set theory[C]//Proceeding of the 4th International Conference Rough Sets and Knowledge Technology. Gold Coast, Australia, 2009: 642–649.

[14] ZHOU Bing, YAO Yiyu, LUO Jigang. Cost-sensitive three-way email spam filtering[J]. Journal of intelligent information systems, 2014, 42(1): 19–45.

[15] ZHANG Hengru, MIN Fan, HE Xu, et al. A hybrid recommender system based on user-recommender interaction[J]. Mathematical problems in engineering, 2015, 2015: 145636.

[16] YAO Jingtao, AZAM Nouman. Web-based medical decision support systems for three-way medical decision making with game-theoretic rough sets[J]. IEEE transactions on fuzzy systems, 2015, 23(1): 3–15.

[17] QIAN Yuhua, ZHANG Hu, SANG Yanli, et al. Multigranulation decision-theoretic rough sets[J]. International journal of approximate reasoning, 2014, 55(1): 225–237.

[18] LI Huaxiong, ZHOU Xianzhong, HUANG Bing, et al. Cost-sensitive three-way decision: a sequential strategy [M]//LINGRAS P, WOLSKI M, CORNELIS C, et al. Rough Sets and Knowledge Technology. Berlin, Germany: Springer, 2013: 325–337.

[19] LIU Dun, LI Tianrui, LIANG Decui. Three-way decisions in dynamic decision-theoretic rough sets[M]//LINGRAS P, WOLSKI M, CORNELIS C, et al. Rough Sets and Knowledge Technology. Berlin, Germany: Springer, 2013: 291–301.

[20] SHE Yanhong. On determination of thresholds in three-way approximation of many-valued nm-logic[M]//CORNELIS C, KRYSZKIEWICZ M, ŚLĘZAK D, et al. Rough sets and Current Trends in Computing. Cham, Germany: Springer, 2014: 136–143.

[21] 于洪. 三支聚类分析[J]. 数码设计, 2016, 5(1): 31–35, 30.
YU Hong. Three-way cluster analysis[J]. Peak data science, 2016, 5(1): 31–35, 30.

[22] 于洪, 毛传凯. 基于 k-means 的自动三支决策聚类方法[J]. 计算机应用, 2016, 36(8): 2061–2065, 2091.
YU Hong, MAO Chuankai. Automatic three-way decision clustering algorithm based on k-means[J]. Journal of computer applications, 2016, 36(8): 2061–2065, 2091.

作者简介：

吴俊伟，男，1993 年生，硕士研究生，主要研究方向为云计算。

姜春茂，男，1972 年生，教授，硕士生导师，主要研究方向为云计算、嵌入式计算和大数据。主持省部级以上科研项目 3 项，厅局级项目 5 项，省级教改项目 2 项。发表 SCI、EI 检索文章 30 余篇。

2019 年第四届控制与机器人工程国际会议（ICCRE 2019）
2019 The 4th International Conference on Control and Robotics Engineering (ICCRE 2019)

Welcome to the official website of the 4th International Conference on Control and Robotics Engineering (ICCRE 2019). The conference will be held in Nanjing, China during April 20—23, 2019. The aim as well as objective of ICCRE 2019 is to present the latest research and results of scientists related to Control and Robotics Engineering topics. ICCRE2019 is Sponsored by Hohai University, China and Jiangsu Key Laboratory of Power Transmission and Distribution Equipment Technology.

An engineering discipline that is on the rise, robotics engineering is a breeding ground for creativity and innovation from people with a background in mechanical, electrical, or software engineering. Robotics engineers may work in the agricultural, military, medical, and manufacturing industries, among others, conceiving of new uses for robots, designing improved robots for existing systems, or repairing and maintaining industrial robots, says the Princeton Review. Because robots are already widely used (on production lines, for example), hands-on technical jobs can easily be found in the robotics engineering field, but there are also plenty of opportunities to take on more inventive roles in experimental arenas.

会议网站：www.iccre.org

会议日期：2019 年 4 月 20—23 日

会议地点：中国南京