计算机科学
COMPUTER SCIENCE

基于三支聚类的云任务优化调度

马新宇, 姜春茂, 黄春梅

引用本文

马新宇, 姜春茂, 黄春梅.基于三支聚类的云任务优化调度[J]. 计算机科学, 2022, 49(11A): 211100139-7.

MA Xin-yu, JIANG Chun-mao, HUANG Chun-mei. Optimal Scheduling of Cloud Task Based on Three-way Clustering [J]. Computer Science, 2022, 49(11A): 211100139-7.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.211100139

基于三支聚类的云任务优化调度

马新宇[1]　姜春茂[2]　黄春梅[2]
1 黑龙江财经学院财经信息工程学院　哈尔滨 150500
2 哈尔滨师范大学计算机科学与信息工程学院　哈尔滨 150025
(1136760479@qq.com)

摘　要　云平台是支撑当今诸多高新技术发展的重要基础设施之一。作为云计算系统体系规划的一个重要组成部分,调度技术直接关系着云计算组成系统中的任务完成时间和能耗问题。为保证基础设施及服务模式下的云任务高效调度,提出了一种基于三支聚类的云任务优化调度算法(Three-Way Clustering Optimal scheduling Programming,TWOCP)。针对云任务属性的多样化特点,结合三支聚类算法对重叠任务和模糊任务进行粒化,并依次调度每个类簇的核心域任务和边界域任务;通过结合动态规划算法对粒化任务进行优化调度,以期实现最少任务完成时间。Cloudsimplus 实验仿真结果表明,所提算法可以降低任务完成时间和能源消耗,有效保障云数据中心的可用性。

关键词:三支决策;三支聚类;任务调度;动态规划;云计算
中图法分类号　TP301

Optimal Scheduling of Cloud Task Based on Three-way Clustering

MA Xin-yu[1],JIANG Chun-mao[2] and HUANG Chun-mei[2]
1 College of Finance and Information Engineering,Heilongjiang University of Finance and Economic,Harbin 150500,China
2 School of Computer Science and Information Engineering,Harbin Normal University,Harbin 150025,China

Abstract　Cloud computing is an important infrastructure supporting many high-tech developments. Furthermore,cloud task scheduling technology is directly related to the task completion time and energy consumption in the cloud computing system. In order to ensure the efficient scheduling of cloud tasks in the infrastructure and services mode,this paper proposes a three-way clustering optimal scheduling programming algorithm(TWOCP). According to the diversified characteristics of cloud task attributes,the overlapping and fuzzy tasks are granularly combined with three-way clustering algorithms,and the core region and boundary region tasks of each cluster are scheduled in turn. A dynamic programming algorithm is used to optimize the scheduling of granular-task to minimize the task completion time. Experimental simulation results in Cloudsimplus show that the proposed algorithm can reduce task completion time,energy consumption and effectively guarantee the availability of cloud data center.

Keywords　Three-way decisions,Three-way clustering,Task scheduling,Dynamic programming,Cloud computing

1　引言

近年来,云计算逐渐发展为全球信息通信技术行业的发展核心,随之而来的是云数据中心的巨大能源消耗,预计在未来还会不断增加[1]。因此,如何合理利用资源成为云计算研究的重点[2]。调度技术将云用户所提交的任务匹配到合适的虚拟机,其实现算法的优劣直接影响任务完成时间、系统负载均衡、能源消耗、资源利用率等一系列用户和云服务供应商之间与经济利益相关的性能指标[3-7]。将复杂云任务合理地调度到计算机节点上,实现云任务和资源的合理匹配与按需分配十分重要,因此也吸引了众多专家进行研究。其中,聚类是一种实现云任务调度的有效手段。例如文献[8]使用模糊 c-均值聚类算法对任务进行聚类,实现云资源的合理分配。文献[9]考虑对异构资源进行二次聚类,将云任务匹配到合适的虚拟机,从而减少任务与资源的匹配时间和任务搜索空间。文献[10]提出了面向实时云任务的细粒度任务合并调度算法,根据实时任务的松弛时间将任务分成紧急调度队列、正常调度队列和松弛调度队列进行任务调度。

上述文献中从不同维度分别对任务或资源进行聚类,并针对性地设置相应调度函数,其实验结果也表明聚类调度可显著改善任务执行时间。然而,在实时云任务聚类时并不全是非此即彼的类别划分,往往存在中间模糊任务集合,在聚类中出现云任务重叠现象[11],在调度时将会产生较高的能耗代价。三支聚类[12-14]借鉴三支决策基本思想[15-18],使用两个集合表示一个类簇,其核心域和边缘域对具有多标签的对象进行进一步划分,可以有效解决重叠任务问题。文献[19]尽管利用三支聚类算法对任务进行聚类划分,然而在后续的评分调度过程中,其追求最大化资源的使用,忽略了任务的完成时间和能量消耗。因此,本文考虑将三支聚类引入云任务调度问题,实现云任务的聚类划分。

基金项目:黑龙江省自然科学基金(LH2020F031)
This work was supported by the Natural Science Foundation of Heilongjiang Province,China(LH2020F031).
通信作者:姜春茂(hsdrose@126.com)

在任务调度策略的研究中，文献[20-21]分别在聚类的基础上，使用 Min-Min 算法和 Max-Min 算法相结合的选择调度算法进行任务和资源的合理匹配。但是 Min-Min 算法和 Max-Min 算法属于一种局部贪心策略，前者优先选取完成时间最短的任务进行调度，而后者优先选取完成时间最长的任务进行调度。这两种算法容易陷入局部最优情况，并不能保证全局最优。因此本文结合动态规划算法进行任务调度，可以有效避免遗传算法、贪心算法等易陷入局部最优的缺点，寻找全局最优调度策略。追求最小化任务完成时间，把任务与虚拟机的匹配看成多阶段决策的组合优化，进行任务调度策略。

本文第 2 节简单回顾了三支聚类、动态规划算法以及调度模型的相关工作；第 3 节提出一种基于三支聚类的云任务优化调度算法，结合动态规划思想对粒化任务进行优化调度，构建了一种基于 TWOCP 的优化调度策略；第 4 节进行了相关仿真实验对比和结果分析；最后总结全文并展望未来。

2 相关工作

2.1 三支聚类算法

基于粗糙集理论中概念的正域、负域和边界域的语义解释，在决策粗糙集中，Yao 等提出三支决策理论[22-23]：从正域生成的规则代表接受某事物；从负域生成的规则代表拒绝某事物；从边界域生成的规则代表无法做出接受或拒绝的判断，即：延迟决策。其基本思想是：

假设 $S=(U,R)$ 为一个信息系统，其中 $U=\{x_1,x_2,\cdots,x_n\}$ 为有限非空集合的对象；R 为定义在 U 上的一种二元关系。基于关系 R，三支决策通过函数 f 将对象集 U 划分成 3 个互不相交的区域，针对不同的区域设置相应的解决策略。

$$f:U\rightarrow\{R_1,R_2,R_3\} \tag{1}$$

其中，$R_1,R_2,R_3\subseteq U$，$U=R_1\cup R_2\cup R_3$；$R_1\cap R_2=\varnothing$，$R_2\cap R_3=\varnothing$，$R_1\cap R_3=\varnothing$。针对 3 个区域 R_1，R_2 和 R_3，采取 3 种不同的策略：S_1，S_2 和 S_3。随着科学研究的不断深入，三支决策理论逐步显现并得到广泛应用。

三支聚类是三支决策与聚类模型结合的产物[24-25]，其扩展了对象与集合的隶属关系。讨论对象集合 $U=\{x_1,x_2,\cdots,x_n\}$ 为一个非空对象集合，其中 x_i 是一个对象，且具有 d 维属性，即 $x_i=\{x_i^1,x_i^2,\cdots,x_i^d\}$，$i\in[1,n]$。在三支聚类视角下，对象和类存在 3 种关系：

(1)对象确定属于一个类；

(2)对象确定不属于一个类；

(3)对象可能属于也可能不属于一个类。

因此提出三支的表示形式：

$$C=\{Co(C),Fr(C)\} \tag{2}$$

其中，$Co(c)\subseteq U$ 且 $Fr(c)\subseteq U$，设 $Tr(c)=U-Co(c)-Fr(c)$。故 $Co(c)$，$Fr(c)$，$Tr(c)$ 构成了类簇的 3 个域，即核心域、边缘域和琐碎域。

$$\begin{aligned}&CoreRegion(C)=Co(C)\\&FrigeRegion(C)=Fr(C)\\&TrivialRegion(C)=U-Co(C)-Fr(C)\end{aligned} \tag{3}$$

若 $x\in Co(C)$，则对象 x 确定属于类 C；若 $x\in Tr(C)$，则对象 x 确定不属于类 C；若 $x\subseteq Fr(C)$，则对象 x 可能属于类 C。

2.2 动态规划算法

动态规划[26]是在研究多阶段决策过程的优化问题时提出的优化原理，也就是将原问题转换为一系列相互联系的单一问题，寻求全局最优解。目前，动态规划算法被广泛应用在云计算任务调度[27]和移动机器人路径规划研究[28]等。

在动态规划中，每个阶段的状态变化，将会导致这个阶段的决策发生变化。首先确定阶段变量，对于 n 个阶段问题记为 $k=1,2,\cdots,n$；其次确定状态变量 S_k，并确定状态变量的集合；然后确定决策变量 x_k 以及决策集合 $D_k(s_k)$，其状态转移函数表示为：

$$s_{k+1}=T_k(s_k,x_k) \tag{4}$$

最后根据已知条件得到指标函数 $F_{k,n}$，$f_k(s_k)$，$F_{1,n}$，$f_1(s_1)$，根据最优化原理得到式(5)：

$$f_k(s_k)=\{F_{k,n}(s_k)\}=opt\{d(s_k,x_k)+f_{k+1,n}(s_{k+1})\} \tag{5}$$

其中，opt 根据实际情况写最大值或最小值，于是得到基本方程：

$$\begin{cases}f_k(s_k)=opt\{d(s_k,x_k)+f_{k+1,n}(s_{k+1})\},k=n,n-1,\cdots,1\\f_{n+1}(s_{n+1})=0\end{cases} \tag{6}$$

2.3 调度模型

任务调度是云计算中重要的组成部分之一，在云计算中，任务调度问题可以表述为用户提交任务调度到合适的虚拟机，调度策略将直接影响整个云计算中任务的完成时间、系统的资源利用率、能源消耗、系统负载均衡和云数据中心的可用性等问题[29-30]。任务调度策略还需满足用户与云服务商的约束条件，如服务等级协议(Service-Level Agreement)和服务质量(Quality of Service)等。云任务调度架构如图 1 所示。

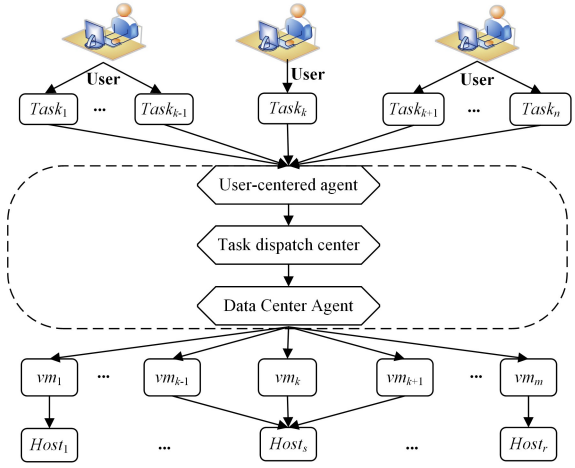

图 1 云任务调度架构

Fig. 1 Cloud task scheduling architecture

如图 1 所示，上述任务调度架构可分为 3 步：

Step1 用户根据自己的需求提交任务 $Task$，任务被添加到任务队列中，任务调度器通过调度技术获取以虚拟机为主的可用资源信息；

Step2 根据调度策略将任务分配到合适的虚拟机 vm 上执行；

Step3 任务完成后进行汇总，反馈给用户。

3 基于三支聚类的云任务优化调度模型

本节根据云任务属性的多样化，使用三支聚类算法对用

户所提交的任务进行聚类划分,并提出一种基于三支聚类的云任务优化调度算法。

3.1 调度框架

在云计算系统中,对云平台用户提交的任务历史数据进行分析,包括数据结构、数据属性和数据量等,在系统数据预处理中,筛除无关数据及属性,并将有用的原始数据进行规范化。设定用户提交的任务集合 $T=\{task_1,task_2,\cdots,task_n\}$ 由 n 项独立的任务组成,其中 $task_i=\{task_i^{ip},task_i^{mips},task_i^{mem},task_i^{bw}\}$,$i\in[1,n]$分别由任务标识 $task_i^{ip}$、请求的计算 $task_i^{mip}$、内存 $task_i^{mem}$ 和网络带宽 $task_i^{bw}$ 组成的任务 $task_i$。云计算资源的基本单元为虚拟机 $V=\{v_1,v_2,\cdots,v_m\}$ 表示由 m 项虚拟机组成计算机资源,其中 $v_j=\{v_j^{ip},v_j^{mips},v_j^{size},v_j^{bw}\}$ 表示由标识 v_j^{ip}、计算能力 v_j^{mip}、存储能力 v_j^{size} 和网络传输能力 v_j^{bw} 组成的虚拟机 v_j。任务调度是集群的核心,其负责任务的调度与资源的合并和迁移。本文建立的调度模型如图 2 所示。

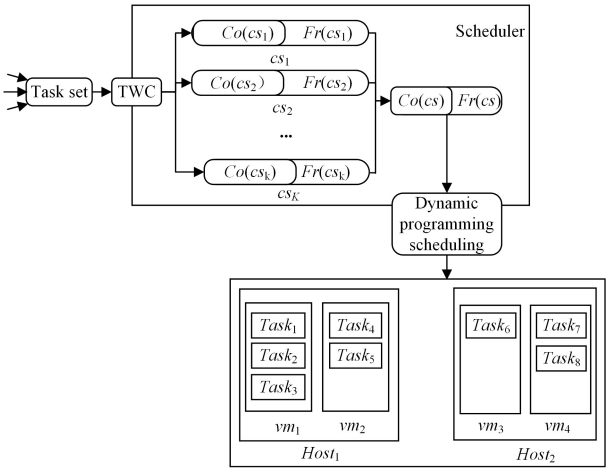

图 2 调度模型

Fig. 2 Scheduling model

如图 2 所示,用户提交的任务 $task$ 进入任务队列 T,使用三支聚类算法(TWC)将用户提交的任务划分成 K 个类簇 $(cs_1,cs_2,\cdots,cs_k)$,然后使用动态规划算法依次调度每个类簇的核心域任务和边界域任务,为用户提交的任务选择合适的虚拟机 vm 执行。

3.2 基于三支聚类的云任务粒化(TWC-TG)

采用三支聚类算法对云任务集合 T 进行聚类划分,提出云任务类簇的三支表示形式如下:

$$cs_i=\{Co(cs_i),Fr(cs_i)\} \tag{7}$$

其中,$Co(cs_i)\subseteq T$ 且 $Fr(cs_i)\subseteq T$,设 $Tr(cs_i)=T-Co(cs_i)-Fr(cs_i)$,则 $Co(c_i)$,$Fr(c_i)$,$Tr(c_i)$ 构成了类簇 cs_i 的核心域、边界域、琐碎域。其满足如下性质:

$$\begin{aligned}&T=Co(cs_i)+Fr(c_i)+Tr(c_i)\\&Co(cs_i)\cap Fr(c_i)=\varnothing\\&Co(c_i)\cap Tr(c_i)=\varnothing\\&Fr(c_i)\cap Tr(c_i)=\varnothing\end{aligned} \tag{8}$$

其语义解释如下:若任务 $task\in Co(cs_i)$,则任务 $task$ 属于类簇 cs_i,并将其划分到类簇 cs_i 的 Co 域中;若任务 $task\in Fr(cs_i)$,则任务 $task$ 可能属于类簇 cs_i,并将其划分到类簇 cs_i 的 Fr 域中;若任务 $task\in Tr(cs_i)$,则任务 $task$ 确定不属于类簇 cs_i,并将其划分到类簇 cs_i 的 Tr 域中。其聚类结果为 $\{(Co(cs_1),Fr(cs_1)),(Co(cs_2),Fr(cs_2)),\cdots,(Co(cs_k),Fr(cs_k))\}$三支聚类算法主要包括寻找最佳聚类数与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确定最佳聚类结果。

(2)确定类簇内的对象,首先通过近邻域确定类簇 Fr 域对象,然后使用差值排序进一步划分类簇 Co 域和 Fr 域对象。其具体内容为:对于类簇 cs_k 和其类簇中心 $centroid_k$,任务 t_i,t_j 与 t_i 的近邻域 $Neig_q(t_i)$,其中 $Neig_q(t_i)$ 表示在欧氏距离上离 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 t_i 为 $Fr(cs_h)$ 任务,然后依次计算类簇 cs_h 内任务 $t_1,t_2,\cdots,t_l$ 与类簇中心 $centroid_k$ 的距离,并从小到大排序,根据插值排序法,找到第一个距离差值最大的任务对 t_{p-1} 和 t_p,将任务 $t_1,t_2,\cdots,t_{p-1}$ 划分到 $Co(cs_h)$,将 $t_p,t_{p+1},\cdots,t_l$ 划分到 $Fr(cs_h)$。

基于上述讨论,算法 1 给出基于三支聚类的云任务粒化算法(TWC-TG)的伪代码。

算法 1 TWC-TG 算法

输入:任务集合 T,近邻数 q

输出:CS={(Co(cs_1),Fr(cs_1)),…,(Co(cs_k),Fr(cs_k))}

```
1. while( k<=√N) do
2.    classList<-choose(k,T);
3.    while(changeCenter(classList)) do
4.       ComputeDistance(T,classList);
5.       update(classCenter);
6.    endwhile
7.    k<-k+1;
8. endwhile;
9.    for each i∈n
10.   Partation(ti,Fr,CS);/* 根据 q 近邻将任务划分到边缘域 */
11. end for
12. for each i∈(n-Fr.length)
13.   Partation(tq,Fr,CS);/* 插值排序将任务划分到边缘域 */
14. end for
15. return CS
```

3.3 基于 TWC-TG 的优化调度方法

基于上述 TWC-TG 聚类算法,得到 TWOCP 云任务优化调度算法。对粒化后的任务采用动态规划调度算法,依次调度每个类簇的核心域与边缘域的任务。其借鉴动态规划的基本思想,考虑将云任务与虚拟机的匹配问题描述为多阶段决策问题,即将可选择的策略限定于一定范围,目的就是在这个范围内找到任务完成时间最短的策略。

虚拟机处理当前分配给它的所有任务的时间为 vpt,如 $vpt(j)$ 表示第 j 虚拟机执行完当前所分配的所有任务的时间。当 n 个不同的任务 $task$ 调度到 m 个虚拟机 vm 上的期望运行时间 ***ETC***(Excepted Time to Compute)是一个 $n\times m$ 的矩阵。其中第 i 行表示任务 i 在每个虚拟机 vm 上的预测执行时间;第 j 列表示在第 j 虚拟机上各任务的预测执行时间。其中,$etc_{i,j}$ 表示第 i 任务调度到第 j 虚拟机上的预测执行时间。

$$\boldsymbol{ETC}=\begin{pmatrix} etc_{01} & \cdots & etc_{1m-1} \\ \vdots & \ddots & \vdots \\ etc_{n-11} & \cdots & etc_{n-1m-1} \end{pmatrix}_{n\times m} \tag{9}$$

在任务集合 T 中，设任务集合 T 的最优调度为 G，所需的调度时间为 $T(N,t)$，$S=\{T-T_i\}$ 是 T 的任务子集，任务 i 是在第 j 虚拟机上调度的第一个任务，当任务开始执行时，所需的时间为 t_i，若其他任务需被执行，则需等待此任务完成后才可以运行。因此 G 代表了全部 T 任务的最优调度，需要的时间为 $\min\{T(i-1,t_{i-1}),T'\}$，其中 T' 为任务集合 S 所需的调度时间，其中 $S=\{N-G(T(i-1))\}$，则 $T'=T\{S,\min(T(i-1))\}$。

由以上任务调度最优子结构可以得出：

$$T(N,0)=\min\{\min T_i(i,0),T(N-(i),\min_{j=0}^{m-1}\max vpt(j))\}0\leqslant i\leqslant n-1 \tag{10}$$

其中，$\min\max_{j=1}^{m-1} vpt(j)$ 为前 i 项任务调度后所花费的时间。由此可以得出如式(11)所示的递归式：

$$T(S,t)=\min\{\min etc_{ij}+T(S-\{i\}),\max\{t-\min_{j=0}^{m-1}vpt(j)\}\} \tag{11}$$

其中，$\max\{t-\min_{j=0}^{m-1} vpt(j)\}$ 为任务调度完成 i 任务后，剩下的任务在虚拟机上还需要的时间。

基于上述递归式，本文提出了动态规划的云任务调度算法，算法主要有以下 5 个步骤：

Step1(阶段变量) 用户提交 n 个任务，将阶段按转化次数划分 $t=1,2,\cdots,n$。

Step2(状态变量) 对于 n 个任务调度到 m 个虚拟机的问题，定义每个阶段可选择的虚拟机状态变量为 $S_t(1,2,3,\cdots,m)$，因为每个虚拟机在每个阶段中都可被选择，故每个阶段可选择的状态集合为 S_t。

Step3(决策变量) x_k 表示第 $i(1,2,\cdots,n)$ 阶段的工作，x_i 为调度给第 i 项任务的虚拟机；当第 i 任务调度到第 j 个虚拟机时，x_j 取值为 1，否则 x_j 取值为 0。

Step4(允许决策集合) $X_k(k=1,2,\cdots,n)$ 表示第 k 阶段决策变量可能的取值，故 $X_k=\{1,2,\cdots,m\}$。

Step5(状态转移方程) 第 t 阶段状态变量为 S_t，若该阶段的决策变量 X_k 确定，则 $t+1$ 阶段状态变量 S_{t+1} 的值也可以确定，即 $S_{t+1}=\Delta t(S_t,X_t)$，$S_{t+1}$ 随着 S_t 和 X_t 的变化而变化，此为 t 阶段到 $t+1$ 阶段的状态转移规律。根据题目描述为求 $(X_1,X_2,\cdots,X_m)$，使得它是下面问题的最优解。

$$\min\max\{\sum_{i=0}^{n-1}X_i etc_{i,0},\sum_{i=0}^{n-1}X_i etc_{i,1},\cdots,\sum_{i=0}^{n-1}X_i etc_{i,m-1}\} \tag{12}$$

基于动态规划算法思想，对于每个任务，依次计算 S，其中每个集合中元素都是一个 m 元组 $(L_1,L_2,\cdots,L_m,X)$，L_i 为第 i 个虚拟机的完成时间；X 为任务分配变量，当 $X_{i,j}=1$ 时表示任务 i 分配给虚拟机 j。

基于上述讨论，算法 2 给出 TWOCP 优化调度算法的伪代码。

算法 2 TWOCP 优化调度算法

输入：CS，m 个虚拟机

输出：taskorderList

```
1. for each i∈n//确定每个类簇内任务的执行顺序
2.    dp[i]<-0;//dp
3.    for each j∈i-1
4.      if T[j]<=T[i] &&. dp[j] > dp[i]
5.           dp[i]<-dp[j];
6.        else
7.        dp[i]<-dp[i]+1;
8.      end if
9.    end for
10.   taskorderList.add( min(dp[i]) )//任务顺序添加
11. end for
12. return taskorderList;
```

4 实验结果与分析

4.1 实验环境及数据集

本文采用云计算仿真软件 CloudSimPlus6.0 进行实验，实验操作系统为 64 位 Window10，CPU 为 i5-7300HQ，内存为 12GB，硬盘为 1T，主频为 3.5GHz。使用的语言为 JAVA，在 IDEA 开发平台上进行实验。

实验数据来自 Google cluster trace 和 PlanetLab[31]，其中 Google cluster trace 数据集中包含用户名字、任务的标识、内存、任务类型、所需的计算资源、资源请求本地磁盘空间和带宽等信息。根据本文的要求，选取了数据表中的任务的标识、内存、所需的计算资源和网络带宽。PlanetLab 数据为 CoMon 工程的一部分，该工程中，负载从数据中心每 5min 收集一次。该负载流是一种典型的 laaS 云环境负载流，如典型的 Amazon EC2。

4.2 评价指标

为了评价任务调度算法的性能，实验采用了任务完成时间 Makespan、能耗和云数据中心的可用性[32] 这 3 个评价指标。选取指标的含义及选取依据如下：

(1)任务完成时间(Makespan)

该指标为用户提交的任务在数据中心计算集群中执行所花费的总时间，即最后一个执行完毕任务的完成时间，完成时间 Makespan 越小说明该调度算法的执行感效率越高，且任务的完成时间将直接影响用户的满意度。

$$Makespan=\max_{j=0}^{m-1}\sum_{i=0}^{n-1}x_i etc_{i,j} \tag{13}$$

其中，n 和 m 分别表示任务个数与虚拟机个数，若第 i 个任务调度到第 t 个虚拟机时，则 $x_i=t$，即 $x_i\in[0,m-1]$。

(2)能耗(Energy Consumption)

该指标表示用户提交的任务完成时，云数据中心物理服务器的总能量消耗。研究表明云数据中心的物理服务器的能耗值主要和其 CPU 在某一时刻的使用率密切相关，基本呈线性关系。

$$S_i(w)=\lambda_i * S_i^{\max}+(1-\lambda_i) * S_i^{\max} * w_i \tag{14}$$

其中，$S_i(w)$ 是物理服务器 i 的总能量消耗，常量 λ_i 表示物理服务器 i 空闲与高负荷时的比率，$S_i^{\max}$ 表示服务器被充分利用时的最大能量消耗，w_i 表示物理服务的 CPU 使用率。物理服务器 i 在 t_i 和 t_j 之间时间段的总体能耗记为 EC_i。

$$EC_i=\sum_{t_i}^{t_j}S(w_i(t_j)) \tag{15}$$

其中，$w_i(t_j)$ 是服务器 i 在 t_j 时刻 CPU 的使用率。$S(w_i(t_j))$ 为服务器 i 在 t_j 时刻的能耗。根据以上信息，我们可以计算

出数据中心 CDC 的总能耗 EC。

$$EC=\sum_{\forall i\in CDC} EC_i \tag{16}$$

(3)云数据中心服务的可用性(Availability)

从用户的角度来说，云数据中心服务可用性也十分重要，因此本文引入云数据中心可用性来衡量用户对云服务的体验。其服务可用性为分配资源与请求资源的比率。

$$Availbility=\frac{\sum(AllocatedMIPS)}{\sum(\mathrm{Re}questdMIPS)}\times 100\% \tag{17}$$

4.3 任务完成时间对比

为了验证 TWOCP 调度算法在任务执行时间的表现，将 TWOCP 与 CTSA-3WD[20]，DPSA[27] 以及 K-C-MM[21] 调度算法进行对比。实验结果如图 3 所示，其中图 3(a)为在 Google cluster trace 数据集上的结果，图 3(b)为使用 PlanetLab 数据的结果。

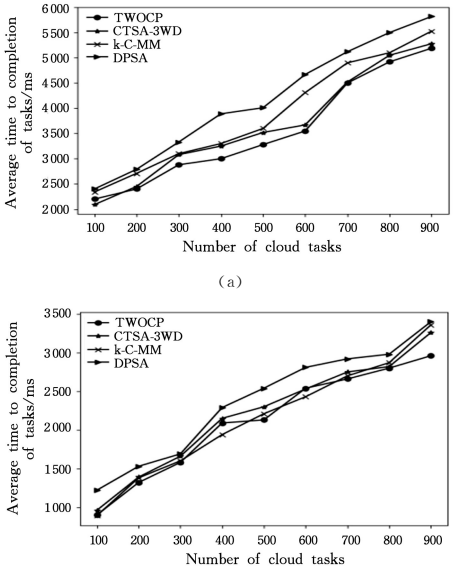

图 3 Makespan 比较

Fig. 3 Makespan comparison

如图 3(a)和(b)所示，相比于 CTSA-3WD，DPSA 和 K-C-MM 算法，TWOCP 调度算法运用了更少的时间完成了相同数量的任务。其中，DPSA 算法尽管同样采用动态规划算法，但它没有考虑任务细粒度化，据统计，对任务进行细粒度划分可以有效减少任务的调度时间，故 DPSA 算法执行任务所花费的时间最长。而 K-C-MM 算法虽然使用聚类算法对任务进行聚类划分，但其在调度算法中使用了 Min-Min 算法与 Max-Min 算法相结合的选择调度算法，动态规划调度算法相比启发式调度算法而言是在全局寻找最优解。CTSA-3WD 采用三支决策思想根据任务属性划分成轻负载任务、重负载任务和轻重负载任务，然而调度时同样采取了 Min-min 和 Max-min 调度算法，相比动态规划算法，更易陷入局部最优解的状态。经统计，TWOCP 算法在任务运行时间上比 CTSA-3WD，DPSA 与 K-C-MM 算法减少约 8%。

4.4 能耗对比

为了比较不同的调度算法下数据中心的总能耗，将 TWOCP，CTSA-3WD，DPSA 和 K-C-MM 算法进行了能量消耗的对比。实验结果如图 4 所示，其中图 4(a)为在 Google cluster trace 数据集上的结果，图 4(b)为采用 PlanetLab 数据的结果。从图 4(a)和(b)中可以看出，随着云任务数量的增加，云数据中心的总能耗也在不断增加，在相同云任务数量的情况下，相比于 CTSA-3WD、DPSA 算法和 K-C-MM 算法，TWOCP 算法在云数据中心所消耗的能源最低，但能耗差很小。其主要原因是 TWOCP 算法将任务调度到虚拟机中描述成多阶段决策问题，为任务选择合适的虚拟机执行，所以本文提出的算法在一定程度上减少了能源消耗。

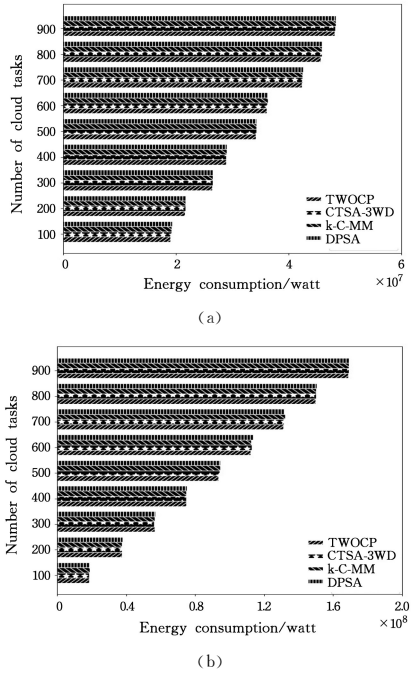

图 4 能量消耗对比

Fig. 4 Comparison of energy comsumption

4.5 云数据中心服务可用性对比

本文在不同任务数量调度的情况下进行了云服务可用性的计算，实验结果如图 5 所示，其中图 5(a)为在 Google cluster trace 数据集上的结果，图 5(b)为采用 PlanetLab 数据的结果。

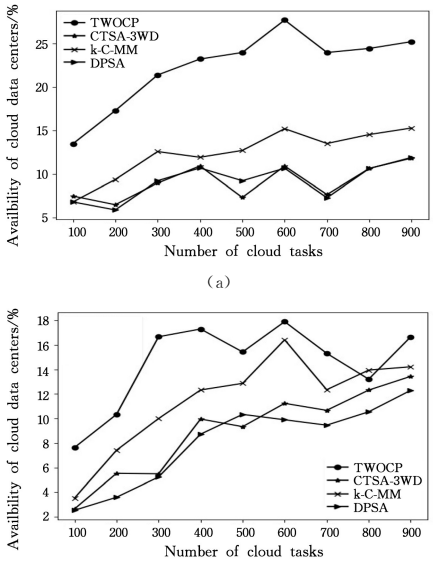

图 5 不同算法的可用性对比

Fig. 5 Comparison of availability of different algorithms

从图 5(a)和(b)中可以看出，在 Google cluster trace 数据集任务较少时，CTSA-3WD，K-C-MM 和 DPSA 这 3 种算法

的的云数据中心的可用性相近。随着云任务增多，K-C-MM算法的云数据中心的可用性稳定增长；TWOCP算法随着云任务的增加云数据中心的可用性基本稳定。相比于CTSA-3WD，K-C-MM和DPSA算法，本文提出的TWOCP算法的云数据中心的可用性最好、最稳定。

基于上述的实验，得出TWOCP，K-C-MM，CTSA-3WD和DPSA调度算法在不同任务数量下的评价指标结果，如表1和表2所列，分别为在Google cluster trace数据集和PlanetLab数据下的评价指标对比。可以看出，本文提出的算法在相同任务数量下的指标基本优于其他算法。

表1 Google cluster trace数据集下评价指标对比

Table 1 Comparison of evaluation indicators on Google cluster trace

	Times				Energy consumption				availability			
	TWOCP	K-C-MM	CTSA-3WD	DPSA	TWOCP	K-C-MM	CTSA-3WD	DPSA	TWOCP	K-C-MM	CTSA-3WD	DPSA
100	2200	2340	2090	2400	**19028759**	19229213	19239243	19339243	**13.49**	6.83	7.50	6.83
200	**2400**	2710	2450	2790	**21660430**	21739416	21730430	21841036	**17.30**	9.38	6.51	5.91
300	**2880**	100	3080	3330	**26486636**	26583266	26583022	26603408	**21.39**	12.59	8.98	9.25
400	**3000**	3300	3250	3890	**28890515**	29006614	28903055	29032640	**23.22**	11.94	10.94	10.71
500	**3280**	3600	3520	4010	**34179351**	34314773	34293132	34316504	**23.97**	12.72	7.34	9.25
600	**3550**	4310	3670	4670	**36103840**	36395423	36251944	36231341	**27.70**	15.22	10.93	10.68
700	**4500**	4900	4520	5120	**42366186**	42488195	42450384	42531842	**23.97**	13.52	7.67	7.30
800	**4920**	5100	5050	5500	**45738807**	45921847	45922911	45917577	**24.44**	14.55	10.66	10.65
900	**5190**	5530	5280	5820	**48148037**	48340508	48242973	48336589	**25.22**	15.30	11.84	11.92

表2 PlanetLab数据集下评价指标对比

Table 2 Comparison of evaluation indicators on PlanetLab

	Times				Energy consumption				availability			
	TWOCP	K-C-MM	CTSA-3WD	DPSA	TWOCP	K-C-MM	CTSA-3WD	DPSA	TWOCP	K-C-MM	CTSA-3WD	DPSA
100	**900**	890	960	1220	**18741981**	18830418	18781734	19032267	**7.64**	3.53	2.65	2.54
200	**1320**	1380	1390	1530	**37483963**	37503854	37486754	37851303	**10.34**	7.42	5.54	3.60
300	**1580**	1600	1660	1690	56593284	56574621	**56225945**	56843457	**16.67**	10.00	5.50	5.24
400	2090	**1940**	2150	2290	**74967309**	74967926	74789437	75335266	**17.30**	12.33	9.96	8.73
500	**2130**	2210	2300	2540	**93709908**	94607732	94438470	94744588	**15.43**	12.87	9.34	10.34
600	2540	**2430**	2530	2810	**112032345**	112451890	113036290	113922622	**17.90**	16.42	11.24	9.90
700	**2660**	2700	2750	2920	**131193871**	132295891	131561211	131928551	**15.32**	12.35	10.67	9.46
800	**2800**	2870	2820	2980	**149935835**	150303193	150282862	150670533	13.21	**13.95**	12.32	10.55
900	**2960**	3360	3260	3400	**168677835**	169250801	169045174	169412514	**16.66**	14.22	13.44	12.30

结束语 本文针对云任务调度问题，提出三支聚类动态规划调度算法，首先根据任务的属性特征，使用三支聚类算法对任务进行聚类划分，然后采用动态规划算法对粒化后的任务进行调度，以减少全部任务完成时间为目标，将任务和虚拟机的匹配看成多阶段决策过程。与CTSA-3WD，DPSA和K-C-MM调度算法的对比实验结果表明：本文提出的算法降低了任务的完成时间，减少了能源消耗，且有效保障了云数据中心的可用性。

未来工作中，将加大数据集，引入大数据处理方法，对任务更加细化聚类，同时加入云环境中的其他因素，如服务质量、任务之间的关联程度等，在此基础之上进一步优化调度算法。

参 考 文 献

[1] BETH W, DEBORAH A, AMIP S, et al. Assessing the environmental impact of data centres part 1: Background, energy use and metrics[J]. Building and Environment, 2014, 82: 151-159.

[2] WEI Y, BLAKE M B. Service-Oriented Computing and Cloud Computing: Challenges and Opportunities[J]. IEEE Internet Computing, 2010, 14(6): 72-75.

[3] ARORA N, BANYAL R K. An Overview of Traditional and Intelligent Task Scheduling Algorithms in a Cloud Computing Environment[C]// International Conference on Intelligent Machines(ICIM'19). 2019.

[4] MAHMOOD I, SADEEQ M, ZEEBAREE S, et al. Task Scheduling Algorithms in Cloud Computing: A Review[J]. Turkish Journal of Computer and Mathematics Education (TURCOMAT), 2021, 12(4): 1041-1053.

[5] R DAHAN F, HINDI K E, GHONEIM A, et al. An adapted antinspired algorithm for enhancing Web service composition[J]. Nternational Journal on Semantic Web & Information Systems, 2017, 13(4): 181-197.

[6] WANG P W, DING Z J, JIANG C J, et al. Automatic web service composition based on uncertainty execution effects[J]. IEEE Transactions on Services Computing, 2016, 9(4): 551-565.

[7] FU X. Task Scheduling Scheme Based on Sharing Mechanism and Swarm Intelligence Optimization Algorithm in Cloud Computing[J]. Computer Science. 2018, 45(S1): 303-307.

[8] LIU J Z, SUN B, ZHU C G. Application of fuzzy C-means algorithm in task scheduling problem[C]// The 10th Annual Conference of China Institute of Communications. 2014: 310-313.

[9] LI J L, DING D, LI T. Multi-objective hybrid cloud task scheduling using twice clustering[J]. Journal of Zhejiang University (Engineering Science), 2017, 51(6): 1233-1241.

[10] JIANG C M, WANG K X. Real-time cloud task energy-saving scheduling algorithm based on three queues [J]. Journal of Zhengzhou University(Natural Science Edition), 2019, 51(2): 66-71.

[11] JIAO P, YU H. Overlapping Types in Soft Clustering[J]. Journal of Kunming University of Science and Technology(Natural Science Edition), 2015, 40(3): 64-69.

[12] YU H. Tree-way Cluster Analysis [J]. Peak Data Csioence, 2016, 5(1): 31-35.

[13] YU H,CHU S,YANG D. Autonomous Knowledge-oriented Clustering Using Decision-Theoretic Rough Set Theory[C]// Rough Set & Knowledge Technology-international Conference. DBLP,2012.

[14] HONG Y,LIU Z,WANG G. An automatic method to determine the number of clusters using decision-theoretic rough set[J]. Acoustic Bulletin,2014,55(1pt. 2):101-115.

[15] WEN P,LI Y,POLKOWSKI L,et al. Three-Way Decision:An Interpretation of Rules in Rough Set Theory[C]// International Conference on Rough Sets & Knowledge Technology. Berlin:Springer-Verlag,2009:642-649.

[16] GUO D D,JIANG C M,YANG L. An effectiveness measure approach for movement-based three-way decision model[J]. Journal of Chinese Computer Systems,2021,42(12):2511-2518.

[17] YAO Y. The superiority of three-way decisions in probabilistic rough set models[J]. Information Sciences,2011,181(6):1080-1096.

[18] GUO D D,JIANG C M. Multi-stage Regional Transformation Strategy in Move-based Three-way Decisions Model[J]. Computer Science,2019,46(10):279-285.

[19] WU J W,JIANG C M. Load-aware score scheduling of three-way clustering for cloud task[J]. CAAI Transactions on Intelligent Systems,2019,14(2):316-322.

[20] WANG Z,JIANG C M. Cloud Task Scheduling Algorithm Based on Three-way Decisions[J]. Computer Science,2021,48(S1):420-426.

[21] LIU Y L,TAO Y,CHEN Z F,et al. Research on the Selective Task Scheduling Algorithm Based on K-maens[J]. Journal of Chang University of Science and Technology(Natural Science Edition),2019,42(5):109-115.

[22] YAO Y Y. Three-Way Decisions and Cognitive Computing[J]. Cognitive Computation,2016,8:543-554.

[23] GUO D D. The TAO Model and Acting Measures of Three-way Decision under Granular Computing Perspective[D]. Harbin:Harbin Normal University,2021.

[24] YAO Y. Tri-level thinking:models of three-way decision[J]. International Journal of Machine Learning and Cybernetics,2020,11(5):947-959.

[25] YU H,MAO C K. Automatic three-way decision clustering algorithm based on k-means[J]. Journal of Computer Applications,2016,36(8):2061-2065.

[26] LARSON R,CASTI J. Principles of Dynamic Programming[M]. Dekker M,1982.

[27] SHI S F,LIU Y B. Cloud computing task scheduling research based on dynamic programming[J]. Journal of Chongqing University of Posts and Telecommunications(Natural Science Edition),2012,24(6):687-692.

[28] ZHOU P,WANG Z M,LI Z N,et al. Complete Coverage Path Planning of Mobile Robot Based on Dynamic Programming Algorithm[C]// Proceedings of the 2nd International Conference on Electronic & Mechanical Engineering and Information Technology (EMEIT 2012). 2012.

[29] NI L,SUN X,LI X,et al. GCWOAS2:Multiobjective Task Scheduling Strategy Based on Gaussian Cloud-Whale Optimization in Cloud Computing[J]. Computational Intelligence and Neuroscience,2021,2021:1-17.

[30] ZHNAG K Q,TU Z Y,CHU D H,et al. Survey on Service Resource Availability Forecast Based on Queuing Theory[J]. Computer Science,2021,48(1):26-33.

[31] PARK K S,PAI V S. CoMon:A mostly-scalable monitoring system for PlanetLab[J]. Acm Sigops Operating Systems Review,2006,40(1):65-74.

[32] SONG J,PAN H. DDBS:a Data Dependency Based Virtual Machine Selection Strategy for Cloud Data Centers[J]. Journal of Chinese Computer Systems,2020,41(2):350-355.

MA Xin-yu, born in 1996, postgraduate. His main research interests include cloud computing three-way decision and so on.

JIANG Chun-mao, born in 1972, Ph.D, professor, is a member of China Computer Federation. His main research interests include cloud computing and big data, intelligent data decision making.