一种面向不可分任务需求和部署约束的动态多维资源公平分配机制

李杰[1, 2]，汪建洲[1]
（1. 昆明理工大学信息工程与自动化学院，云南 昆明 650504；
2. 昆明理工大学云南省计算机技术应用重点实验室，云南 昆明 650504）

摘　要：如何公平、高效地将多维资源分配给需求变化的用户是云计算资源共享的关键问题。该场景下的动态资源分配通常面临着用户任务最小粒度资源需求难以再被分割、任务需求与服务器配置不匹配等问题。现有资源公平分配机制多基于用户任务需求无限可分或任务执行与服务器配置均匹配的理想前提，难以保证分配可行。通过深入分析时变不可分任务需求和任务部署约束的特点，设计了一种基于累计任务份额公平的时变任务份额公平分配机制，以保证资源分配的公平性和效率。理论结果表明，该机制满足激励共享、相差一个任务资源的无嫉妒和帕累托最优属性。基于阿里云数据集的实验结果表明，与现有的公平分配机制相比，该方法有效地减少了用户的等待、作业排队和作业完成时间。

关键词：动态多维资源分配；不可分任务需求；任务部署约束；累计任务份额公平
中图分类号：TP301.6
文献标志码：A
doi：10.11959/j.issn.2096−3750.2024.00447

A fair multi-resource allocation mechanism for time-varying discrete jobs with placement constraints

LI Jie[1, 2], WANG Jianzhou[1]
1. Faculty of Information Engineering and Automation, Kunming University of Science and Technology, Kunming 650504, China
2. Yunnan Key Laboratory of Computer Technology Application, Kunming University of Science and Technology, Kunming 650504, China

Abstract: A key issue in resource sharing in cloud computing is how to fairly and efficiently allocate the multi-resources to users with dynamic demand. Multi-resource fair allocation in a cloud computing system usually faces problems, such as subdividing the minimum granularity of users' resource requirements, and the mismatch between task requirements and server configurations. Most of the existing mechanisms for multi-resource fair allocation are based on the ideal assumption that the task demands of user are infinitely divisible or that the task execution and server configuration are matched, which makes it difficult to guarantee that the allocation is feasible. By analyzing the characteristics of time-varying indivisible task demands and task placement constraints, a time-varying task share fairness allocation mechanism based on cumulative task share fairness was designed to ensure the fairness and efficiency of resource allocation. Theoretical analysis shows that the TV-TSF mechanism satisfies the sharing incentive, envy-freeness up to one item, and Pareto optimal properties. Simulation results based on the Alibaba cluster dataset show that, compared with the existing fair allocation mechanisms, the TV-TSF mechanism proposed can effectively reduce the waiting time, job queuing time, and job completion time of users.

收稿日期：2024−10−15；**修回日期：**2024−12−09
通信作者：李杰，lijacademic@126.com
基金项目：国家自然科学基金资助项目（No. 62362043，No. 62262034）；云南省计算机应用技术重点实验室开放基金资助项目（No. 2022201）；昆明理工大学人培基金资助项目（No. KKZ3202403168）
Foundation Items: The National Natural Science Foundation of China (No. 62362043, No. 62262034), The Yunnan Key Laboratory of Computer Technologies Application (No. 2022201), The Project of Talent Training of Kunming University of Science and Technology (No. KKZ3202403168)

Key words: dynamic multi-resource allocation, indivisible task demand, task placement constraint, cumulative task share fairness

0　引言

云计算资源提供商将云计算资源动态地提供给用户使用，由于其可靠和灵活的特点，目前大量的企业和个人都倾向于将计算和储存业务提交至云计算平台上执行。资源公平分配是一种非常有效的云计算资源分配方式，它可以高效地将低负载的用户没有使用的资源分配给高负载用户使用，有效地促进了系统的负载均衡并提高了资源利用率。

云计算用户提交的需求通常是动态变化的。Alibaba 2021的集群数据显示[1]，“双十一”购物节期间的数据处理任务量，明显比平时的任务量大。这些任务需求通常由多个具有相同资源需求的实例组成[2-5]，每个实例多维资源需求（如对CPU、内存的需求）及提交时间、运行时间等都会不断变化。通常，每个实例都很难再被分割，如执行一个大型计算任务实例需要2 CPU core和2 GB内存，如果只分配给用户1 CPU core和1 GB内存，则将导致这个实例执行失败。另外，随着云计算中心硬件的更新换代，几代服务器部署在同一个数据中心的情况很常见[6-7]。当云计算用户提交它们不可分的任务需求时，这些差异化的不可分任务资源需求如果与硬件不兼容，将导致任务部署约束。例如，做图形处理的任务实例需要部署在有GPU资源的服务器上，而进行数据处理的任务需要部署在具有大量内存的服务器上。谷歌集群数据统计显示，在谷歌云计算集群中有超过50%的用户任务面临任务部署约束[6]。在面向动态不可分割的任务和任务部署约束的云计算资源公平分配机制设计中，如果处理不当将很难保证系统中用户执行效率，并导致系统资源利用率低。

因此，如何在以上实际云计算中心环境中公平高效地分配资源，以有效保证系统中用户任务执行效率，并提高整个集群的资源利用率，具有重要的应用价值。目前，对于多维资源分配机制的公平性和高效性主要由经济学中的一些性质来定义，当满足这些性质时，可以保证公平高效的资源分配[8-9]。

1) 激励共享（SI, sharing incentive）[10]：系统中用户获得通过资源共享方式执行的任务数不少于每个用户通过独享方式执行的任务数。

2) 无嫉妒（EF, envy-freeness）[11]：系统中任意用户都不偏好其他用户的分配。

3) 相差一个任务需求的无嫉妒（EF1, envy-freeness up to one item）[12]：对于任意两个用户，拿走其中一个用户执行一个任务所需资源后，另一个用户将不再嫉妒该用户。

4) 帕累托最优（PO, Pareto optimal）[13]：增加系统中任意用户执行任务数，必将使其他用户分配的资源量减少。

针对云计算资源公平分配，目前最经典的多维资源公平分配机制是占优资源公平（DRF, dominant resource fairness）[14]，其核心思想是将每个用户资源需求中占所有资源需求比例最大的资源分配最大最小化。DRF满足了前文定义的1)、2)和4)的公平性质，并广泛应用于Mesos和Hadoop YARN等计算框架中。尽管DRF和后续研究工作[15-19]都致力于多维资源公平分配，但这些机制都没有明确地考虑任务部署约束。

针对任务部署约束下资源公平分配的问题，Ghodsi等[20]提出了一种单资源最大最小公平分配机制（CMMF, constrainted max-min fairness）来保证任务部署约束下单资源分配的公平性。Binois等[21]基于Kalai-Smorodinsky讨价还价方法（KS, Kalai-Smorodinsky bargaining solution）设计了任务份额公平分配机制（TSF, task share fairness）[22]来解决任务部署约束下多资源公平分配问题。考虑进程间公平调度，Zhou等[23]设计了Enuomia机制来保证任务部署约束下进程调度的公平性。而在云边协同资源公平分配方面，文献[24]提出了一种考虑任务压缩传输中带宽资源消耗的多资源公平分配机制。尽管以上资源分配机制能够保证任务部署约束下分配的公平性，但这些机制通常假设任务资源需求粒度可以无限分割，且用户任务提交不随时间改变的静态情形，很难适用于解决具有动态不可分任务的多资源公平分配问题。

针对动态需求下资源公平分配问题，文献[25-26]分别提出了两种考虑累计公平的资源分配方法。考虑用户动态到达和离开的情形，文献[27]提出了一种公理化的网络资源分配公平性理论，该理论有

效地统一并扩展了现有的公平性度量，为理解和实现更加公平和高效的资源分配机制提供了新的视角和方法。文献 [10]基于最大最小份额公平提出了一种考虑任务执行时长等信息的资源近似公平分配机制，有效地提高了分配效率。Fikioris 等[28]对动态需求情形下最大最小公平分配和占优资源公平分配进行拓展，使其能够满足动态资源近似公平。

综上，在多维资源公平分配、任务部署约束下资源公平分配和动态资源公平分配问题中，目前的研究已经取得了许多进展，但是同时考虑动态不可分资源和任务部署约束情形的资源分配问题仍然面临着挑战。

本文旨在研究如何在用户时变不可分任务和任务部署约束条件下，保证多种资源分配的公平和高效。其中，用户的时变任务指的是用户提交的需要执行的任务所需资源量和需要完成的任务数会动态变化，任务不可分指的是分配给用户的资源所能执行的最小粒度任务为非负整数个；任务部署约束表示用户的任务是否能够在某一台服务器上运行。本文与现有研究比较见表 1。

针对动态不可分任务需求和任务部署约束下时变多资源分配问题，本文设计了一种基于累计任务份额最大最小的多资源公平分配机制，并证明该机制满足激励共享、帕累托最优和相差一个任务需求的无嫉妒属性。

为实现时变任务份额公平（TV-TSF, time-varying task share fairness）分配机制，本文设计了一个 TV-TSF 累计任务份额最大最小分配算法，可以有效提高系统资源利用率，进而减少用户任务等待、排队和完成时间。

1　问题模型描述

1.1　时变任务约束公平分配问题描述

本文主要符号描述见表 2。假设在云计算资源共享系统中有 M 台服务器，服务器的集合定义为 $S=\{1,2,\cdots,M\}$，系统中的服务器有 K 种资源，资源种类的集合定义为 $R=\{1,2,\cdots,K\}$，如 CPU、内存和硬盘资源等。对于服务器 $m\in S$，该服务器资源归一化的资源容量为 $\boldsymbol{C}_m=(C_{m1},C_{m2},\cdots,C_{mK})^{\mathrm{T}}$，其中，$C_{mr}$ 表示服务器 m 第 r 种资源的容量占所有服务器这种资源总量的比值。本文将每种资源的总容量归一化为 1，即

$$\sum_{m=1}^{M}C_{mr}=1,\forall r\in R \tag{1}$$

整个系统在一系列的时间槽 $[1,T]=\{1,2,\cdots,T\}$ 内运行，其中，T 为一个已知常数。系统中用户的集合定义为 $U=\{1,2,\cdots,N\}$，由于用户提交的任务通常是由许多具有相同资源需求的任务组成的，用户 $i\in U$ 在时间槽 $t\in[1,T]$ 提交的任务需求表示为 $\theta_i(t)=(\boldsymbol{d}_i(t),B_i(t),e_i,\boldsymbol{p}_i)$，其中，$\boldsymbol{d}_i(t)=(d_{i1}(t),d_{i2}(t),\cdots,d_{iK}(t))^{\mathrm{T}}$ 表示在时间槽 t 用户 i 提交的单个任务的资源需求向量，$d_{ir}(t)$ 为用户 i 的每个任务对第 r 种资源的需求量与第 r 种资源总量的比值。$B_i(t)$ 为在时间槽 t 用户 i 提交的需要执行的任务数。e_i 为用户 i 提交的任务执行时间。在云计算资源共享系统中，系统升级换代等原因导致服务器配置各不相同，所以用户的任务不可能在所有服务器上运行。考虑用户任务部署限制，根据文献[22]，对于用户 i，令 $\boldsymbol{p}_i=(p_{i1},p_{i2},\cdots,p_{iM})^{\mathrm{T}}$ 为用户 i 的任务部署约束向量，其中，$p_{im}=1$ 表示用户 i 的任务可以

表 1　　**本文与现有研究比较**

资源分配方法	任务部署约束	资源分配模型	任务是否可分	任务执行时长	满足公平性质
DRF	未考虑	静态	可分	未考虑	SI, PO, EF
DRFH	未考虑	静态	可分	未考虑	PO, EF
CMMF	未考虑	静态	可分	未考虑	—
CDRF	未考虑	静态	不可分	未考虑	SI, PO, EF1
TSF	考虑	静态	可分	未考虑	SI, PO, EF
Eunomia	考虑	静态	可分	未考虑	—
HMRF	未考虑	动态	不可分	未考虑	SI, PO
SDRF	未考虑	动态	可分	未考虑	SI, PO, EF
Matching-BagFilling	未考虑	动态	不可分	考虑	SI, PO, EF1
Dynamic MMF	未考虑	动态	不可分	考虑	SI, PO, EF1
本文方法	考虑	动态	不可分	考虑	SI, PO, EF1

在服务器m上执行，而$p_{im}=0$表示用户i的任务不能在服务器m上执行。为了方便起见，假设用户任务部署限制不随时间发生改变。由于实际情况中用户任务资源需求通常大于0，因此，本文假设用户i在时间槽t时，$B_i(t)>0$且$d_{ir}(t)>0$。

表2　　本文主要符号描述

符号	描述
$[1,T]$	系统运行的总时间槽
S	服务器的集合
R	资源种类的集合
$\boldsymbol{C}_m$	服务器m资源归一化的资源容量
C_{mr}	服务器m中第r种资源的容量与所有服务器中该资源总量的比值
U	用户的集合
$\theta_i(t)$	用户i在t时刻的任务需求
$\boldsymbol{d}_i(t)$	用户i在t时刻单个任务的资源需求
$d_{ir}(t)$	用户i在t时刻单个任务的第r种资源需求
$B_i(t)$	用户i在t时刻的任务数
e_i	用户i提交的任务执行时间
$\boldsymbol{p}_i$	用户i任务部署约束向量
p_{im}	用户i与边缘服务器m的任务部署约束
$\boldsymbol{A}_{im}$	用户i在服务器m上的资源分配向量
$\boldsymbol{A}(t)$	所有用户在t时刻的资源分配矩阵
$\boldsymbol{A}_i(t)$	用户i在t时刻的资源分配矩阵
$x_i(t)$	用户i在t时刻在所有服务器中能执行的最大任务数
$x_{im}(t)$	用户i在t时刻在服务器m上能执行的任务数
$s_i(t)$	用户i在t时刻在任务部署约束下共享资源执行的累计任务数与不考虑部署约束时独占所有服务器资源下完成的带权重任务数之比
$\omega_i(t)$	用户i在t时刻任务的重要程度
$h_i(t)$	用户i在t时刻不考虑部署约束并独占所有服务器资源时能够执行的任务数
$k_i(t)$	用户i在t时刻在其专有服务器集合S_i上执行的任务数

对于用户i和服务器m，令$\boldsymbol{A}_{im}=(A_{im1}(t),A_{im2}(t),\cdots,A_{imK}(t))^{\mathrm{T}}$为用户$i$在服务器$m$上的资源分配向量，其中，$A_{imr}(t)$表示在时间槽$t\in[1,T]$，用户$i$在服务器$m$上分配到的第$r$种资源量。令用户$i$在时间槽$t$的资源分配矩阵为$\boldsymbol{A}_i(t)=(\boldsymbol{A}_{i1}(t),\boldsymbol{A}_{i2}(t),\cdots,\boldsymbol{A}_{iM}(t))$，而系统中所有用户在时间槽$t$的资源分配矩阵为$\boldsymbol{A}(t)=(\boldsymbol{A}_1(t),\boldsymbol{A}_2(t),\cdots,\boldsymbol{A}_N(t))$。由于资源容量的限制，考虑用户任务执行时长为$e_i$，若在时间槽$t$所有用户在服务器$m$上分得的第$r$种资源量之和不大于该服务器的这种资源的剩余容量，则表示该分配可行，即$\sum_{i=1}^{N}A_{imr}(t)\leqslant C_{mr}$，$\forall r$，$m$，$t$对用户$i$，假设任务是不可分的，如最少完成的任务为1个。那么在资源分配向量$\boldsymbol{A}(t)$已知的情况下，根据里昂惕夫偏好[16]，用户i在时间槽t在服务器m上能执行的任务数为$x_{im}(t)=\min\{\max\{\alpha\in\mathbf{N}\cup\{0\}:\forall r\in R,A_{im}(t)\geqslant\alpha\cdot d_{ir}(t)\},B_i(t)\}$。考虑任务部署约束，用户$i$在时间槽$t$在所有服务器中能执行的最大任务数为

$$x_i(t)=\min\left\{\sum_{m=1}^{M}x_{im}(t)\cdot p_{im},B_i(t)\right\}\tag{2}$$

特别地，如果用户i在时间槽t分配的资源完成的任务数小于其提交的需求数$B_i(t)$时，剩下的任务将作为其新的需求在后续时间槽自行提交。对于一个有效的资源分配，不应该给用户分配执行任务之外的额外资源，造成资源浪费。所以，根据文献[29]，对于用户i和服务器m，如果减少任何已经分配给用户i的资源将导致用户不能在服务器m上执行原来那么多的任务，则这个分配不浪费，也就是在$\boldsymbol{A}$分配下用户在时间槽t完成的任务数为$x_{im}(A_{im}(t))$，如果减少任何一种资源分配$\tilde{A}_{imr}(t)<A_{imr}(t)$，则$x_{im}(\tilde{A}_{im}(t))<x_{im}(A_{im}(t))$。若不特别指出，本文研究的问题为不浪费情形。

1.2　累计任务份额定义

为解决多时间槽情形下用户部署受限的时变任务的资源公平分配问题，结合文献[22]，本文定义用户i的累计任务份额为：到时间槽T为止，用户i在任务部署约束条件下，通过共享资源能够执行的累计任务数与用户在独占所有服务器资源并不考虑任务部署限制的情况下，累计完成的带权重的任务数之比如式(3)所示。

$$s_i(T)=\frac{\sum_{t=1}^{T}x_i(t)}{\sum_{t=1}^{T}\omega_i(t)\cdot h_i(t)}\tag{3}$$

其中，$\omega_i(t)$为用户i的权重，表示用户i在时间槽t任务的重要程度。$h_i(t)$为用户i不考虑任务部署约束并独占所有系统服务器资源的条件下在时间槽t能够执行的任务数，$C_{mr}(t)$为该情形下时间槽t的资源$\forall r\in R$的剩余容量，即

$$h_i(t)=\sum_{m}^{M}\min_{r}\left\lfloor\frac{C_{mr}(t)}{d_{ir}(t)}\right\rfloor,\ \forall t\in[1,T]\tag{4}$$

1.3　资源分配性质定义

在云计算资源共享系统中，资源分配的公平性

是非常关键的。只有当每个系统中用户能够被公平分配到资源时，多资源共享分配才可行。文献[4]中引入了以下4个重要属性：激励共享、帕累托最优、无嫉妒和可信。结合本文所研究的用户具有多时间槽的有限任务资源需求的情形，将上述公平性描述如下。

1.3.1　激励共享

在多时间槽情况下，系统中所有用户在共享服务器资源时，如果执行的累计任务数不少于其在原来自己所专有的服务器所能执行的累计任务数，则认为该机制满足激励共享属性。假设每个用户 $i \in U$ 在时间槽 $t \in [1,T]$ 专有的服务器集合 $S_i \subset S$ 上执行的任务数为 $k_i(t)$，则如果一个分配 $\boldsymbol{A}$ 满足式(5)，则称分配 $\boldsymbol{A}$ 满足激励共享属性，该属性能够激励用户共享资源。

$$\sum_{t=1}^{T} k_i(t) \leqslant \sum_{t=1}^{T} x_i(\boldsymbol{A}_i(t)),\ \forall i \in U \tag{5}$$

1.3.2　相差一个任务需求的无嫉妒(EF1)

在多时间槽情况下，若所有用户在没有完成任务需求时，至多只能利用别的用户分配的资源多执行一个的带权累计任务数，则认为该机制满足不超过一个任务需求资源的无嫉妒属性。假设对于任意两个不同的用户 i，$j \in U$，如果一个分配 $\boldsymbol{A}$ 满足式(6)，则称分配 $\boldsymbol{A}$ 满足相差一个任务需求的无嫉妒属性，该属性保证了分配的公平性。

$$\sum_{t=1}^{T} \frac{x_i(\boldsymbol{A}_j(t)) - 1}{\omega_j(t)} \leqslant \sum_{t=1}^{T} \frac{x_i(\boldsymbol{A}_i(t))}{\omega_i(t)},\quad \forall i \neq j, t \in [1,T] \tag{6}$$

1.3.3　帕累托最优

若系统中任何用户获得更多的资源，必将使其他用户分配到的资源减少，从而使其他用户能够执行的任务数减少，则认为该机制满足帕累托最优属性。如果在一个机制下的分配 $\boldsymbol{A}$ 使用户 i 执行的累计任务数为

$$\sum_{t=1}^{T} x_i(\boldsymbol{A}_i(t)) \tag{7}$$

且不存在另外一个可行的分配 $\tilde{\boldsymbol{A}}$ 对于所有用户满足

$$\sum_{t=1}^{T} x_i(\boldsymbol{A}_i(t)) \leqslant \sum_{t=1}^{T} x_i(\tilde{\boldsymbol{A}}_i(t)) \tag{8}$$

同时，对一些用户 $i \in U$ 满足

$$\sum_{t=1}^{T} x_i(\boldsymbol{A}_i(t)) < \sum_{t=1}^{T} x_i(\tilde{\boldsymbol{A}}_i(t)) \tag{9}$$

则称该机制满足帕累托最优属性，该属性能够保证高效的资源利用率。

2　TV-TSF机制设计

针对用户在多个时间槽提交多组任务的资源分配问题，本文设计了时变任务份额公平（TV-TSF, time-varying task share fairness）机制解决该问题。TV-TSF采用累计最大最小公平分配，使累计任务份额最小的用户的累计任务份额最大。TV-TSF可描述为

$$\max s(T) \tag{10}$$

$$\text{s.t.} \sum_{t=1}^{T} \frac{1}{h_i(\tau) \cdot \omega(\tau)} \cdot \sum_{t=1}^{T} \sum_{m=1}^{M} x_{im}(t) \cdot p_{im} \geqslant s(T),\quad \forall i \in N \tag{10a}$$

$$\sum_{i=1}^{N} \sum_{\tau=t-e_i+1}^{t} x_{im}(\tau) \cdot d_{ir}(\tau) \leqslant C_{mr},\ \forall m \in M, r \in R,\quad t \in [1,T] \tag{10b}$$

$$x_{im}(t) \in \mathbf{N} \cup \{0\},\quad \forall i \in N, m \in M, t \in [1,T] \tag{10c}$$

其中，目标式(10)和约束式(10a)表示使最小的累计任务份额最大，约束式(10b)保证分配可行，约束式(10c)中的变量定义域为不小于0的整数。

为实现TV-TSF的分配机制，设计了面向不可分任务的TV-TSF累计任务份额最大最小分配算法。其中，在任意一个时间槽 $t \in [1,T]$，进行多轮资源分配使累积任务份额最小的用户的累计任务份额最大。

2.1　TV-TSF分配算法

TV-TSF主算法框架如算法1所示。

算法1　TV-TSF主算法框架

输入　C, d, p, ω

输出　用户在每台服务器上执行的任务数 $\{x_{im}(t)\}$

初始化 $\boldsymbol{C}_m = (C_{m1}, C_{m2}, \cdots, C_{mK})$，$t = 1$，$A = \{1, 2, \cdots, N\}$，$\{x_{im}(t)\} = 0$；

根据式(3)计算 $\{s_i(t)\}$；

while 时间槽 t 还有任务到达 **do**

TV-TSF(t, $\{\boldsymbol{C}_m\}$, $\{\boldsymbol{d}_i, \boldsymbol{p}_i, \omega_i(t)\}$)；

end while

在时间槽 $t \in [1,T]$ 的第一轮分配中，首先判断使累计任务份额最小的用户能够再分配资源使其完成至少一个任务，如果可以，则分配一个任务所需资源给该用户，直到用户由于任务部署限制或者任务需求数的限制不能再增加为止，此时这些任务份额不能再增加的用户，将变为不活跃状态，它们的任务份额在时间槽 t 的剩余多轮分配中将不再改变。接下来在时间槽 t 的第二轮分配中，继续增加处于

活跃状态的用户中累计任务份额最小的用户的任务份额，直到有的用户变为非活跃状态为止。当所有用户均变为非活跃状态时，TV-TSF在时间槽t的多轮分配停止，未完成任务需求的用户，在后面的时间槽将剩余任务作为新任务自行提交。另外，对于时间槽$t \in [1, T]$，本文定义$A^{T(t)}$为第$T(t)$轮开始时的活跃用户集合，$s_i^{T(t)}$定义为第$T(t)$轮结束时用户i的累计任务份额。TV-TSF单时间槽公平分配算法，ALLOC资源分配算法分别用算法2和算法3。

算法2　TV-TSF单时间槽公平分配算法

输入　t，$\{\boldsymbol{C}_m\}$，$\{\boldsymbol{d}_i, \boldsymbol{p}_i, \omega_i(t)\}$

输出　用户在每台服务器上的执行任务数$\{x_{im}(t)\}$

初始化$T(t)=1$，$A^{T(t)}=\{1,2,\cdots,N\}$；

$A^{T(t)}$ =FREEZE(t, $T(t)$, $\{\boldsymbol{C}_m\}$, $A^{T(t)}$, $\boldsymbol{s}^{T(t)}$, $\{x_i(t), \boldsymbol{d}_i, \boldsymbol{p}_i, \omega_i(t)\}$)；

while $A^{T(t)} \neq \varnothing$ **do**

　　$D^{T(t)} \leftarrow \{ i \in A^{T(t)} | \forall j \in A^{T(t)}, s_i^{T(t)-1} \leqslant s_j^{T(t)-1} \}$

　　$i \leftarrow D^{T(t)}$中任意用户；

　　ALLOC(i)；

　　$A^{T(t)}$=FREEZE(t, $T(t)$, $\{\boldsymbol{C}_m\}$, $A^{T(t)}$, $\boldsymbol{s}^{T(t)}$, $\{x_i(t), \boldsymbol{d}_i, \boldsymbol{p}_i, \omega_i(t)\}$)；

　　$T(t)=T(t)+1$；

end while

return $\{x_{im}(t)\}$

算法3　ALLOC资源分配算法

输入　用户i，$\{\boldsymbol{C}_m\}$

输出　$\{x_{im}(t)\}$，$\boldsymbol{s}^{T(t)}$，$\{\boldsymbol{C}_m\}$

m_{opt} = **FIND_BESTFIT_SERVER**($\{\boldsymbol{d}_i, \boldsymbol{p}_i\}$，$\{\boldsymbol{C}_m\}$)；

if $m_{\text{opt}} \neq$ none **then**

　　$\boldsymbol{C}_m = \boldsymbol{C}_m - \boldsymbol{d}_i$；

　　更新$s_i^{T(t)}$；

end if

return $\{x_{im}(t)\}$，$\boldsymbol{s}^{T(t)}$，$\{\boldsymbol{C}_m\}$

以上为资源分配子算法的伪代码，每次将最匹配的服务器资源分配给累计任务份额最低的用户使该用户执行的任务数加1。FIND_BESTFIT_SERVER最佳匹配算法如算法4所示。

算法4　FIND_BESTFIT_SERVER最佳匹配算法

输入　$\{\boldsymbol{d}_i, \boldsymbol{p}_i\}$，$\{\boldsymbol{C}_m\}$

输出　m_{opt}

for m in S **do**

　　if $d_i \leqslant C_m$ 且 $p_{im}=1$ **then**

　　　　$H(i,m,t) = \|\boldsymbol{d}_i/d_{i1} - \boldsymbol{C}_m/C_{m1}\|_1$；

　　else

　　　　$H(i,m,t)=$ none；

　　end if

end for

return $m_{\text{opt}} = \arg\min_{m \in S}\{H(i,m,t)\}$

由于涉及多资源与多服务器的匹配问题，本文设计了子算法用于寻找满足部署约束和剩余资源可行的服务器中剩余资源与用户不可分任务资源需求较为匹配的服务器，以提高资源利用率。FREEZE活跃用户辨别算法如算法5所示。

算法5　FREEZE活跃用户辨别算法

输入　$t, T(t), \{\boldsymbol{C}_m\}, A^{T(t)}, s^{T(t)}, \{x_i(t), \boldsymbol{d}_i, \boldsymbol{p}_i, \omega_i(t)\}$

输出　$A^{T(t)}$/TEMP$^{T(t)}$

初始化TEMP$^{T(t)}=\varnothing$；

for $j \in A^{T(t)}$ **do**

　　ALLOC(j, $\{\boldsymbol{C}_m\}$)；

　　if $s_j^{T(t)}$能再增加 **then**

　　　　TEMP$^{T(t)}$ = TEMP$^{T(t)} \cup j$；

　　end if

end for

return $A^{T(t)}$/TEMP$^{T(t)}$

FREEZE算法对活跃用户集合的每个用户进行判断，看其任务份额是否继续增加，如果不增加，则将该用户变为不活跃用户。其中，TEMP$^{T(t)}$为第$T(t)$轮中不活跃的用户集合。

2.2　TV-TSF机制资源分配公平性质分析

定理1　TV-TSF满足激励共享属性

证明　假设每个用户都不共享资源时，用户i在其专有服务器中，在时间槽$t \in [1, T]$执行的任务数为$k_i(t)$。若令$\omega_i(t) = k_i(t)/h_i(t)$，根据TV-TSF定义可知，在不共享资源情形下，用户i累计任务份额为

$$s(T) = \sum_{t=1}^{T} k_i(t) / \sum_{t=1}^{T} h_i(t) \cdot \omega_i(t) = 1, \ \forall i \in N \quad (11)$$

也就是说，到时间槽T，使所有用户的累计任务份额为1的可行分配是存在的。由于TV-TSF机制是使用户的累计任务份额最大最小（Max-min），所以在TV-TSF机制分配下，到时间槽T所有用户的累计任务份额一定不小于1，即在共享情形下用户完成的累计任务数不小于不共享情形，TV-TSF满足激励共享属性，证毕。

引理 1　对于任意两个用户 i 和 j，有以下式子

$$\sum_{t=1}^{T} h_i(t) \geqslant \sum_{t=1}^{T} \rho_{ji}(t) \cdot h_j(t) \tag{12}$$

其中，

$$\rho_{ji}(t) = \min_r \frac{d_{jr}(t)}{d_{ir}(t)},\quad \forall\, t \in [1,T] \tag{13}$$

证明　假设用户 j 独占所有服务器资源并不考虑任务部署约束时，在服务器 $m \in S$ 所完成的累计任务数为 $\sum_{t=1}^{T} h_{jm}(t)$，此时用户 j 在服务器 m 上分得的资源为 $\sum_{t=1}^{T} h_{jm}(t) \cdot d_{jr}(t)$，$\forall r$。若此时使用户 i 执行任务时不考虑任务部署限制，并将用户 j 分得的资源分配给用户 i，用户 i 能在服务器 m 上执行的累计任务数为 $\sum_{t=1}^{T} x_{im}(t)$，因为不考虑任务部署限制，所以 $\sum_{t=1}^{T} x_i(t) = \sum_{t=1}^{T}\sum_{m=1}^{T} x_{im}(t)$，有

$$\begin{aligned}\sum_{t=1}^{T} x_i(t) &= \sum_{t=1}^{T}\sum_{m=1}^{M} \min_r \frac{h_{jm}(t) \cdot d_{jr}(t)}{d_{ir}(t)} = \\ &\sum_{t=1}^{T} \rho_{ji}(t) \cdot h_j(t)\end{aligned} \tag{14}$$

根据 $h_i(t)$ 的定义可知，用户 i 独占所有资源不考虑任务部署限制所执行的任务数一定不小于其在其他可行分配方案下所执行的任务数，所以有

$$\sum_{t=1}^{T} h_i(t) \geqslant \sum_{t=1}^{T} x_i(t) = \sum_{t=1}^{T} \rho_{ji}(t) \cdot h_j(t) \tag{15}$$

证毕。

引理 2　对于任意两个用户 i，j，如果用户 i, j 不满足 EF1 性质，则

$$s_i(T) < s_j(T) \tag{16}$$

证明　根据 EF1 性质的定义可得，如果用户 i 超过一个任务的嫉妒用户 j，则

$$\sum_{t=1}^{T} \frac{x_i(\boldsymbol{A}_i(t))}{\omega_i(t)} \leqslant \sum_{t=1}^{T} \frac{x_i(\boldsymbol{A}_j(t)) - 2}{\omega_j(t)} \tag{17}$$

$$\sum_{t=1}^{T} x_i(\boldsymbol{A}_j(t)) = \sum_{t=1}^{T}\sum_{m=1}^{M} x_{jm}(t) \cdot \min_r \frac{d_{jr}(t)}{d_{ir}(t)} \cdot p_{im}$$

因为 $p_{im} \leqslant 1$，可得

$$\sum_{t=1}^{T} x_i(\boldsymbol{A}_j(t)) \leqslant \sum_{t=1}^{T}\sum_{m=1}^{M} x_{jm}(t) \cdot \min_r \frac{d_{jr}(t)}{d_{ir}(t)} \tag{18}$$

根据累计任务份额的定义可得

$$\sum_{t=1}^{T}\sum_{m=1}^{M} x_{jm}(t) \cdot \min_r \frac{d_{jr}(t)}{d_{ir}(t)} = \sum_{t=1}^{T} x_j(t) \cdot \rho_{jr}(t) \tag{19}$$

证毕。

定理 2　TV-TSF 满足 EF1 属性

证明　令 $x_i(\boldsymbol{A}(t))$ 表示在时间槽 $t \in [1,T]$ 结束时，用户 i 在资源分配矩阵 $\boldsymbol{A}$ 已知的情形下执行时间槽 t 执行的任务数。首先，如果用户 $\forall i \in U$ 到时间槽 T 结束完成的累计任务数等于其累计任务需求，即 $\sum_{t=1}^{T} x_i(\boldsymbol{A}(t)) = \sum_{t=1}^{T} B_i(t)$，或者只比累计任务需求少一个，即 $\sum_{t=1}^{T} x_i(\boldsymbol{A}(t)) = \sum_{t=1}^{T} B_i(t) - 1$，即使用户 i 分得更多资源也只能增加至多一个任务的收益，此时用户 i 不会嫉妒其他用户超过一个任务。所以，本文讨论 $\sum_{t=1}^{T} x_i(\boldsymbol{A}(t)) < \sum_{t=1}^{T} B_i(t) - 1$ 的情形。

当 $\sum_{t=1}^{T} x_i(\boldsymbol{A}(t)) < \sum_{t=1}^{T} B_i(t) - 1$ 时，假设 TV-TSF 不满足无嫉妒属性，有用户 i 嫉妒另外一个用户 $j \neq i$，根据无嫉妒的定义可知

$$\sum_{t=1}^{T} \frac{x_i(\boldsymbol{A}_i(t))}{\omega_i(t)} < \sum_{t=1}^{T} \frac{x_i(\boldsymbol{A}_j(t)) - 1}{\omega_j(t)} \tag{20}$$

由于任务不可分，即

$$\sum_{t=1}^{T} \frac{x_i(\boldsymbol{A}_i(t))}{\omega_i(t)} \leqslant \sum_{t=1}^{T} \frac{x_i(\boldsymbol{A}_j(t)) - 2}{\omega_j(t)} \tag{21}$$

根据任务份额的定义可知，式(21)矛盾。综上所述，不存在这样的时间槽 t，当用户 i 分配资源执行的任务数小于其任务需求时，能够利用其他用户分配的资源执行更多的带权任务，所以 TV-TSF 满足无嫉妒属性，证毕。

定理 3　TV-TSF 满足帕累托最优属性

证明　假设 TV-TSF 不满足帕累托最优属性。假设到时间槽 T，存在至少一个没有执行完任务的用户 i 可以在不减少其他用户执行任务数的基础上，执行更多的任务。由于用户 i 可以不减少任何用户执行的任务数，执行更多的任务，所以根据 TV-TSF 的定义可知，用户 i 一定是具有最小累计任务份额的用户，在这种情形下，累计任务份额最小的用户的累计任务份额可以再增加，然而，这与 TV-TSF 满足累计任务份额最大最小目标矛盾，所以 TV-TSF 满足帕累托最优属性，证毕。

3　实验评估

本节基于阿里云数据集（Alibaba traces V2018）的仿真实验来评估 TV-TSF 的性能。本节将 TV-TSF 与公平分配算法 TSF[22] 和 CDRF[16] 进行对比，其中，

CDRF是一种针对不可分任务的公平分配算法，核心思想是最小化用户最大占优资源份额分配向量的方式保证分配的公平性。为了分析TV-TSF保证服务质量方面的效果，本节通过用户等待时间、用户任务排队时间、用户任务完成时间与这两种算法进行比较。具体来讲，用户的等待时间定义为从用户提交第一个任务到其第一个任务开始执行的时间。用户的任务排队时间定义为从用户的每个任务提交到该任务被执行的时间。最后，用户任务完成时间是指从用户任务到该任务的最后一个任务被执行完毕的时间。

3.1　实验设置

为保证实验的有效性和复现性，本节介绍了大规模实验的配置。

1) 服务器配置。模拟了由100台服务器组成的集群的资源分配，由于阿里云数据集中只包含CPU和内存资源需求数据，本文实验采用CPU和内存资源来代表多资源分配。根据阿里云官网的服务器配置[30]，仿真实验服务器配置参数见表3。

表3　　仿真实验服务器配置参数

服务器数	CPU/cores	内存/GB
5	2	8
5	4	16
10	8	32
10	12	48
10	16	64
10	24	96
20	32	128
20	52	192
5	64	256
5	104	384

2) 用户任务资源需求。本文实验中，用户每个任务资源需求的分布由阿里云数据集中batch_task文件统计得到。为了分析本文分配机制与现有公平分配机制的分配性能，通过采样得到阿里云数据集任务需求分布，再根据该分布随机生成3种不同类型：小型任务（每个作业任务数小于500）、正常任务和大型任务（每个作业任务数大于或等于500），并分析在正常和大型任务比例分别为25%、50%、80%（相应的小型任务比例为75%、50%、20%）情况下的用户不可分任务需求下的实验结果。

3) 用户作业（job）任务数分布。本文实验中，用户每个作业包含的任务数概率分布由阿里云数据集中batch_task文件的instance_num字段数据统计得到。而每个任务的提交时间，在1~600个时间槽内随机确定。

4) 用户任务执行时长。本文实验中，用户任务的最小运行时间为1个时间槽（slot），即每个任务的运行时间为单个时间槽的整数倍。

5) 实验次数。对于每个实验，生成100组实验样本，通过统计多组实验结果来消除随机化的影响。

3.2　实验结果与分析

3.2.1　用户等待时间分析

不同类型任务下的用户等待时间如图1所示，不同曲线分别代表不同算法下用户等待时间（slot）累积分布函数（CDF, cumulative distribution function），其中用户等待时间表示用户提交任务至其首任务开始执行所间隔的时间，该时间越短说明系统响应越快。μ和σ分别代表实验结果的均值和方差。

正常任务需求如图1(a)所示，正常需求下TV-TSF较TSF和CDRF算法的用户等待时间平均减少了至少0.163个时间槽。大任务占比25%如图1(b)所示，对于用户小任务占多数的情况（大任务占比为25%），由于用户负载低，3种算法的用户等待时间都不高，但与其他两种算法相比，TV-TSF仍然将用户等待时间缩短了至少5个时间槽。大任务占比50%如图1(c)所示，对于中等负载情况（大任务占比为50%），TV-TSF用户平均等待时间至少比其他两种算法快了36个时间槽。大任务占比80%如图1(d)所示，当面对重负载情况时（大任务占比为80%），TV-TSF用户平均等待时间至少比其他两种算法快了438个时间槽。

3.2.2　任务排队时间分析

不同类型任务下的任务排队时间如图2所示，为不同算法下用户任务排队时间累积分布。 用户任务排队时间指其每个任务提交至执行所经过的时间间隔，若其越短，则说明每个用户任务需求都能够得到系统及时响应，体现了系统的公平性。正常任务需求如图2(a)所示，正常需求下TV-TSF较TSF和CDRF算法的用户任务排队时间平均减少了至少0.9个时间槽。大任务占比25%如图2(b)所示，与其他两种公平分配算法相比，TV-TSF的用户任务排队时间减少了至少 56个时间槽。大任务占比50%如图2(c)所示，TV-TSF的用户平均任务排队时间比其他两种算法分别快了56个时间槽。最后，

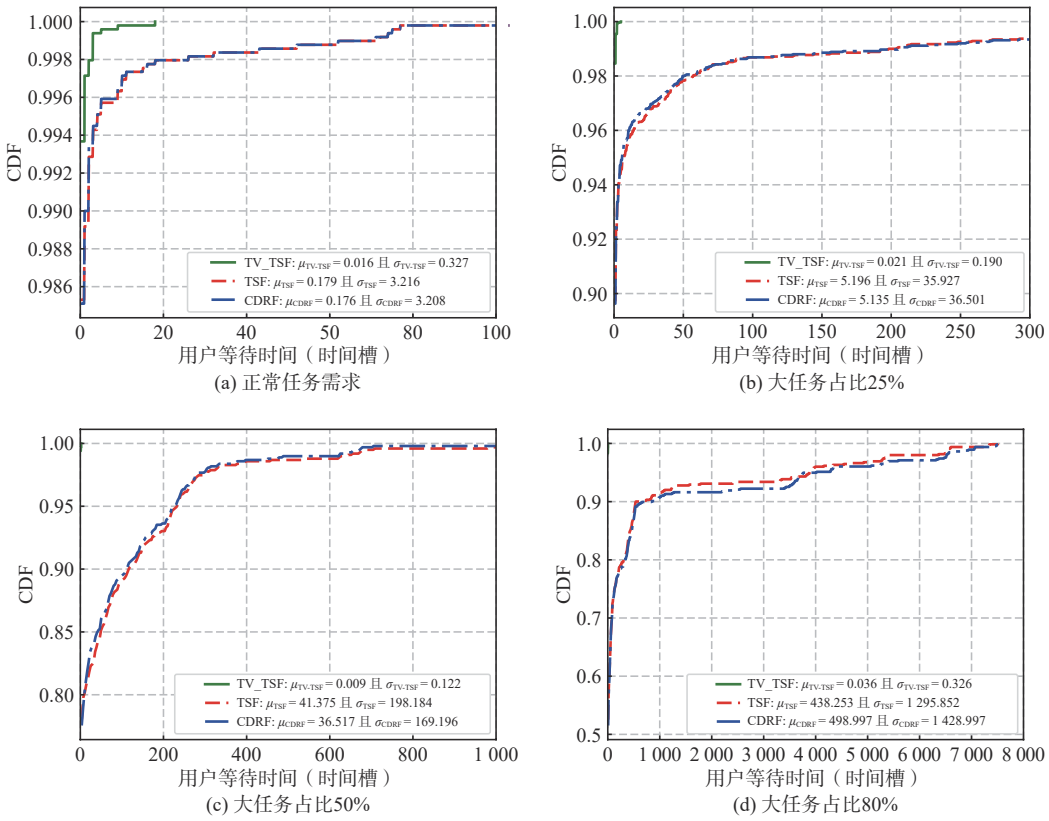


图1　不同类型任务下的用户等待时间

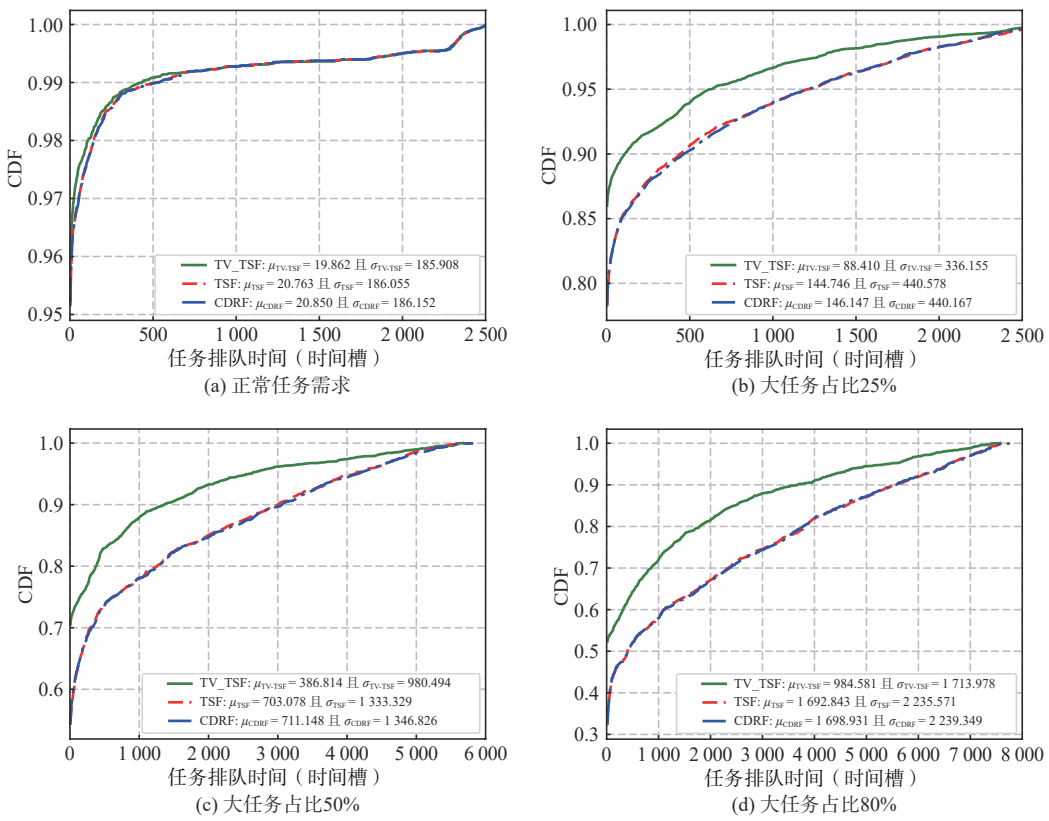


图2　不同类型任务下的任务排队时间

大任务占比80%如图2(d)所示，TV-TSF的用户任务平均排队时间和完成时间比其他两种算法分别快了708个时间槽。可以看出，由于TV-TSF考虑长期公平性，所以负载越大越能保障分配的公平，进而缩短用户任务排队时间。

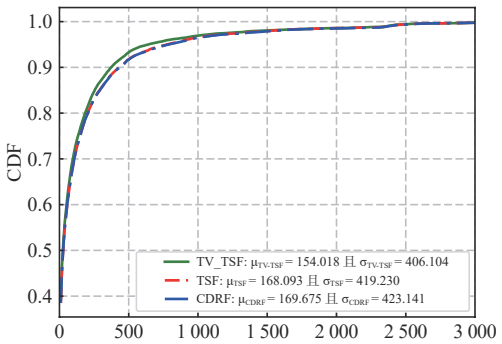
3.2.3　任务完成时间分析

最后，本节对不同需求类型下的任务完成时间进行分析，任务完成时间指用户每个任务完成所需要经历的时间间隔，若此时间越短，说明系统任务执行效率越高。不同类型任务下的任务完成时间如图3所示。

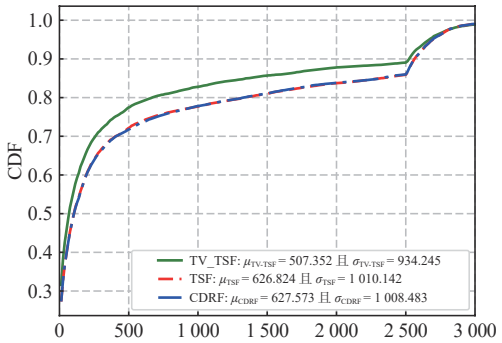
正常任务需求如图3(a)所示，在正常需求下，与其余两种算法相比，TV-TSF在任务完成时间方面，至少平均减少了14个时间槽。大任务占比25%如图3(b)所示，与其他两种算法相比，TV-TSF将用户任务完成时间减少了至少119个时间槽。随着大任务比例的增加，TV-TSF具有更显著的优势。其中，对于中等负载情形（大任务占比为50%），TV-TSF的任务完成时间比其他两种算法分别快384个时间槽。而当大任务占比为80%时，TV-TSF用户平均任务完成时间比其他两种算法分别快了585个时间槽。可以看出，TV-TSF在保证分配公平性的同时，与其他两种算法相比进一步提高了系统任务的执行效率。

4　结束语

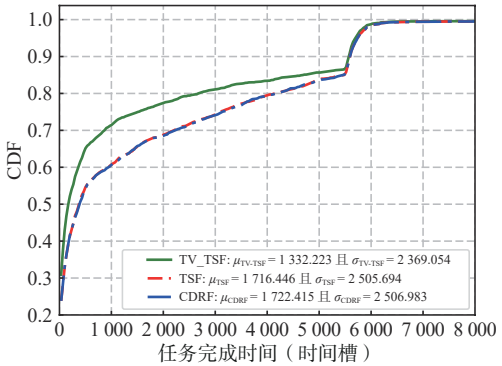
针对时变不可分任务需求和任务部署约束条件下云计算时变资源公平共享分配问题，本文提出了一种基于累计任务份额公平的分配机制TV-TSF，其目标是保证动态情形下资源共享分配的公平性。TV-TSF通过考虑历史累计分配资源记录，保证多时间槽的动态分配更加公平。理论分析表明，TV-TSF能够满足无嫉妒、激励共享、帕累托性质。此外，实验结果表明，与经典的TSF和CDRF公平分配机制相比，TV-TSF能够有效地减少用户等待时间、作业排队时间和作业完成时间，进而更好地保障分配的公平和效率。本文假设的动态环境中，目前只考虑了用户到达后不再离开的场景，忽略了实际问题中用户可能随时到达且可能随时离开的情况。在这类更复杂的场景下，由于用户竞争对手将会实时发生变化，资源分配公平性的定义也将发生改变，本文算法在解决该类实际问题中很难保证资



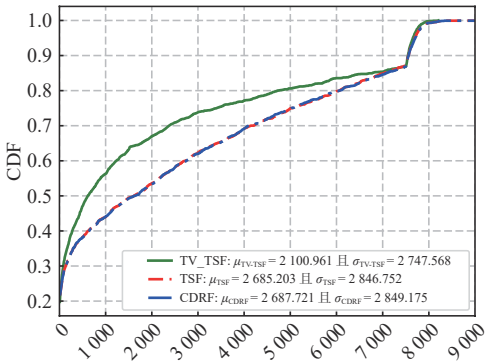
(a) 正常任务需求



(b) 大任务占比25%



(c) 大任务占比50%



(d) 大任务占比80%

图3　不同类型任务下的任务完成时间

源分配实时公平。未来将进一步研究用户可以随时到达和离开的情况下具有任务部署约束的多资源公平分配问题，针对该问题从实时资源分配模型和公平性的定义入手，不断改进资源公平分配算法的实时性和适应性。

参考文献：

[1] AHMED Y N, MOHIDEEN S P. Workload characteristics in cloud data centers: a computational study from alibaba cloud[C]// Proceedings of the 2021 International Conference on Information Science and Communications Technologies (ICISCT). Piscataway: IEEE Press, 2021: 1-5.

[2] GUO H, LI W D. Dynamic multi-resource fair allocation with elastic demands[J]. Journal of Grid Computing, 2024, 22(1): 35.

[3] HINDMAN B, KONWINSKI A, ZAHARIA M, et al. Mesos: a platform for fine-grained resource sharing in the data center[C]// Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation. New York: ACM, 2011: 295-308.

[4] VAVILAPALLI V K, MURTHY A C, DOUGLAS C, et al. Apache Hadoop YARN: yet another resource negotiator[C]//Proceedings of the 4th Annual Symposium on Cloud Computing. New York: ACM Press, 2013: 1-16.

[5] DU L, WO T Y, YANG R Y, et al. Cider: a rapid docker container deployment system through sharing network storage[C]//Proceedings of the 2017 IEEE 19th International Conference on High Performance Computing and Communications; IEEE 15th International Conference on Smart City; IEEE 3rd International Conference on Data Science and Systems (HPCC/SmartCity/DSS). Piscataway: IEEE Press, 2017: 332-339.

[6] SHARMA B, CHUDNOVSKY V, HELLERSTEIN J L, et al. Modeling and synthesizing task placement constraints in Google compute clusters[C]//Proceedings of the 2nd ACM Symposium on Cloud Computing. New York: ACM Press, 2011: 1-14.

[7] 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: ACM Press, 2012: 1-13.

[8] FIKIORIS G, AGARWAL R, TARDOS É. Incentives in dominant resource fair allocation under dynamic demands[M]//SCHÄFER G, VENTRE C, eds. Lecture Notes in Computer Science. Cham: Springer Nature Switzerland, 2024: 108-125.

[9] MESKAR E, LIANG B. Fair multi-resource allocation in heterogeneous servers with an external resource type[J]. IEEE/ACM Transactions on Networking, 2023, 31(3): 1244-1262.

[10] LI B Y. Fair scheduling for time-dependent resources[J]. Advances in Neural Information Processing Systems. New York: ACM Press, 2021, 34: 21744-21756.

[11] ZHANG J X, CHI L X, XIE N, et al. Strategy-proof mechanism for online resource allocation in cloud and edge collaboration[J]. Computing, 2022, 104(2): 383-412.

[12] BEI X H, LI Z H, LUO J J. Fair and efficient multi-resource allocation for cloud computing[C]//Proceedings of the 18th International Conference of Web and Internet Economics. New York: ACM Press, 2022: 169-186.

[13] DENG B, LI W D. Maximin share based mechanisms for multi-resource fair allocation with divisible and indivisible tasks[M]// Communications in Computer and Information Science. Singapore: Springer Nature Singapore, 2022: 263-272.

[14] GHODSI A, ZAHARIA M, HINDMAN B, et al. Dominant resource fairness: fair allocation of multiple resource types[C]//Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation. New York: ACM Press, 2011: 323-336.

[15] DOLEV D, FEITELSON D G, HALPERN J Y, et al. No justified complaints: on fair sharing of multiple resources[C]//Proceedings of the 3rd Innovations in Theoretical Computer Science Conference. New York: ACM Press, 2012: 68-75.

[16] PARKES D C, PROCACCIA A D, SHAH N. Beyond dominant resource fairness[J]. ACM Transactions on Economics and Computation, 2015, 3(1): 1-22.

[17] Gutman A, Nisan N. Fair allocation without trade[C]//Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems. New York: ACM Press, 2012: 719-728.

[18] JOE-WONG C, SEN S, LAN T, et al. Multi-resource allocation: fairness-efficiency tradeoffs in a unifying framework[C]//2012 Proceedings of the IEEE INFOCOM. Piscataway: IEEE Press, 2012: 1206-1214.

[19] LI W D, LIU X, ZHANG X L, et al. Multi-resource fair allocation with bounded number of tasks in cloud computing systems[M]// Communications in Computer and Information Science. Singapore: Springer Nature Singapore, 2017: 3-17.

[20] GHODSI A, ZAHARIA M, SHENKER S, et al. Choosy: max-min fair sharing for datacenter jobs with constraints[C]//Proceedings of the 8th ACM European Conference on Computer Systems. New York: ACM Press, 2013: 365-378.

[21] BINOIS M, PICHENY V, TAILLANDIER P, et al. The Kalai-Smorodinsky solution for many-objective Bayesian optimization [J]. Journal of Machine Learning Research, 2020, 21: 1-42.

[22] WANG W, LI B C, LIANG B, et al. Multi-resource fair sharing for datacenter jobs with placement constraints[C]//Proceedings of the SC '16: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. Piscataway: IEEE Press, 2016: 1003-1014.

[23] ZHOU W, WHITE K P, YU H F. Eunomia: a performance-variation-aware fair job scheduler with placement constraints for heterogeneous datacenters[C]//Proceedings of the 2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS). Piscataway: IEEE Press, 2018: 1034-1039.

[24] LI X X, HU G Q, LI W D, et al. Fair multiresource allocation with

access constraint in cloud-edge systems[J]. Future Generation Computer Systems, 2024, 159: 395-410.

[25] TANG S J, NIU Z J, HE B S, et al. Long-term multi-resource fairness for pay-as-you use computing systems[J]. IEEE Transactions on Parallel and Distributed Systems, 2018, 29(5): 1147-1160.

[26] SADOK H, CAMPISTA M E M, COSTA L H M K. Stateful DRF: considering the past in a multi-resource allocation[J]. IEEE Transactions on Computers, 2021, 70(7): 1094-1105.

[27] LAN T, KAO D, CHIANG M, et al. An axiomatic theory of fairness in network resource allocation[C]//2010 Proceedings IEEE INFOCOM. Piscataway: IEEE Press, 2010: 1-9.

[28] FIKIORIS G, AGARWAL R, TARDOS É. Incentives in dominant resource fair allocation under dynamic demands[M]//Lecture Notes in Computer Science. Cham: Springer Nature Switzerland, 2024: 108-125.

[29] LI W D, LIU X, ZHANG X L, et al. Dynamic fair allocation of multiple resources with bounded number of tasks in cloud computing systems[J]. Multiagent and Grid Systems, 2016, 11(4): 245-257.

[30] Alibaba Group Holding Limited. Alibaba Server Configuration[EB]. 2022.

[作者简介]

李杰（1987– ），男，博士，昆明理工大学信息工程与自动化学院讲师、硕士生导师，主要研究方向为基于算法博弈论的端边云协同资源分配与定价机制设计。

汪建洲（1999– ），男，昆明理工大学信息工程与自动化学院硕士生，主要研究方向为边缘计算与机制设计。