高技术通讯 2021 年 第31 卷 第10 期:1044-1054

doi:10.3772/j.issn.1002-0470.2021.10.005

面向异构物理机的云任务调度策略及性能优化①

范宝芝② 王开宇③ 白小军 金顺福

(燕山大学信息科学与工程学院 秦皇岛 066004)

摘 要 为了在保证云用户响应性能的前提下降低云系统的能源消耗,提出一种移动设备本地处理器持续工作、云端物理机内虚拟机同步休眠、不同物理机间虚拟机异步休眠的任务调度策略。针对云端异构物理机,建立多个同步多重休假的排队模型。利用拟生灭过程和矩阵几何解方法给出系统模型的稳态分布,导出任务平均响应时间和系统平均功率的表达式。实验结果表明,设置任务分配到移动设备本地的概率时,不同性能指标之间存在折衷关系。引入 Logistic 映射混沌机制改进传统的鲸鱼优化算法,给出最优任务分配策略,实现系统成本的最小化。

关键词 云计算;任务调度策略;同步多重休假;平均响应时间;平均功率

0 引 言

随着无线通讯技术的发展,网络应用程序的使用越来越多。移动设备负载能力不足的问题日趋明显,云计算为移动设备的任务卸载提供支持[1]。值得注意的是,不断增长的云应用导致云数据中心消耗着巨大的能源,对环境也产生了恶劣的影响。研究发现,按照每年40% ~60%的增长率估算,到2030 年云数据中心的能源消耗将达到8000 TWh[2]。绿色云计算是社会进步的必然趋势,也是实现可持续发展的前提。缓解云数据中心的高耗能问题是云计算研究的重点内容之一。

文献[3]提出了一种动态迁移虚拟机的调度方法,通过建立一个虚拟机放置/迁移的平台,使用两个调度程序分别调度预期的负载和未预期的负载,有效降低了云数据中心的功耗。文献[4]提出了一种动态的空闲间隔预测方案,该方案可以估计未来中央处理器(central processing unit, CPU)空闲间隔长度,选择考虑成本效益的休眠状态以最小化运行功耗,并提高 CPU 的利用率。以上文献专注于降低云系统的能源消耗,却忽略了云用户对响应性能的需求。

考虑到云用户对响应性能的需求,文献[5]提出了一种基于 (N, T) 休眠机制的云计算中心节能策略。结合唤醒阈值 N 及长度为 T 的休眠计时器,建立多重同步休假随机模型。利用改进飞蛾扑火优化算法,给出了节能策略的联合优化方案,实验证实所提策略在保证用户响应性能的基础上,有效降低了系统能耗。文献[6]提出了一种基于动态阈值的服务器唤醒策略。该策略综合考虑用户端的任务情况和服务器的能耗成本,动态调整任务请求数的阈值,并根据时间优先级唤醒服务器。与静态阈值的服务器唤醒策略相比,该策略有效降低了云计算系统的能耗开销。以上研究专注于云系统本身,未考虑产生任务的移动设备及移动设备自身的处理能力。

本文综合考虑云用户响应性能及云系统的能源消耗,基于部分用户任务在移动设备本地处理器执行的实际情况,提出一种移动设备本地处理器持续

① 国家自然科学基金(61872311,61973261)资助项目。
② 女,1995 年生,硕士;研究方向:计算机网络资源管理;E-mail: 962160562@qq.com
③ 通信作者,E-mail: wangky@ysu.edu.cn
(收稿日期:2021-01-07)

工作和云端物理机内虚拟机同步休眠的任务调度策略。针对云端异构物理机的不同服务速率,在远程云端建立多个带有同步多重休假的 M/M/c_k 排队模型。利用拟生灭过程和矩阵几何解给出系统模型的稳态分布,通过系统实验揭示任务平均响应时间与系统平均运行功率的变化趋势。构造系统成本函数,引入 Logistic 映射混沌机制改进传统鲸鱼优化算法,给出最优任务分配策略,实现系统成本的最小化。

1 任务调度策略及系统模型

1.1 任务调度策略

针对移动设备存储空间不足且处理能力有限等问题,借助于远程云端的帮助,移动设备任务负载能够有效地得到均衡。在任务调度器的协调下,将移动设备产生的任务分为两部分,一部分任务分配到本地处理器接受服务,另一部分任务则卸载到远程云端的虚拟机上接受服务。

远程云端部署多个异构物理机,物理机的集合表示为 $S = \{S_1, S_2, \cdots, S_n\}$,$|S| = n$。通过虚拟化技术在物理机 $S_k(k = 1, 2, \cdots, n)$ 上部署 $c_k(c_k \geqslant 1)$ 个云虚拟机。假设部署在同一台物理机上的虚拟机同构,不同物理机上的虚拟机异构[7]。给出由本地移动设备和远程云端组成的体系结构,如图 1 所示。

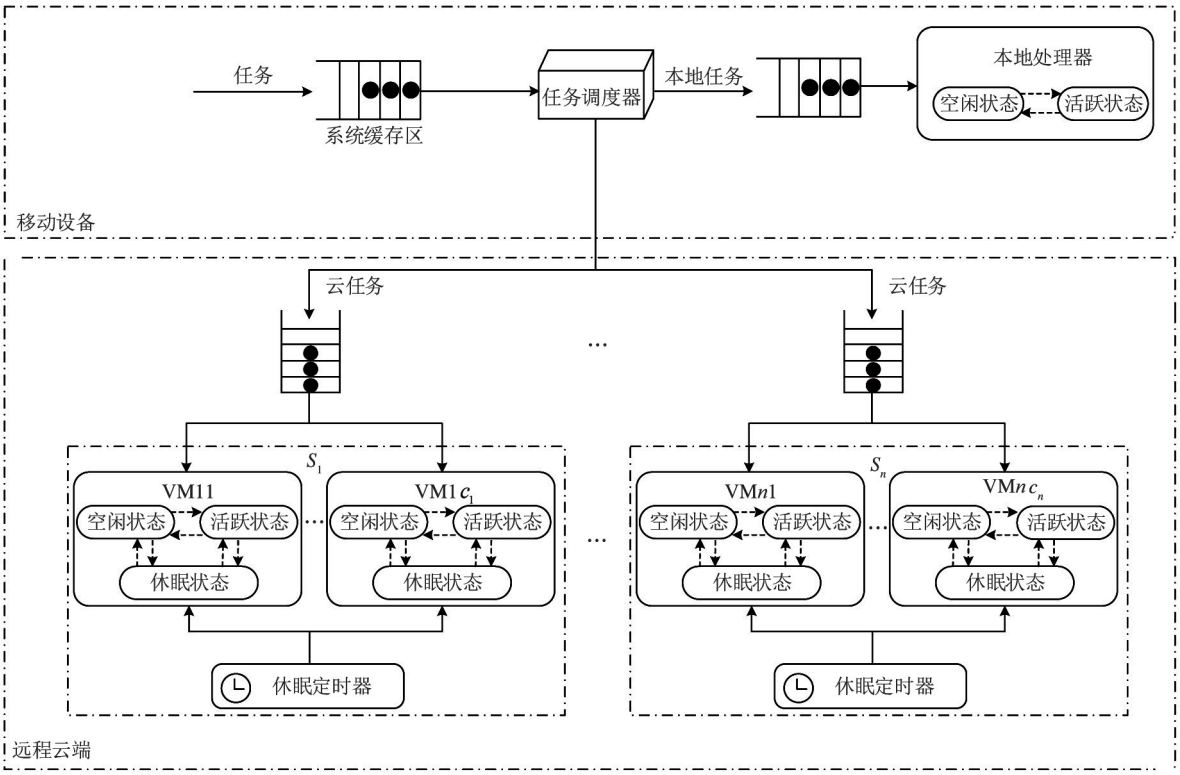


图 1 移动设备与远程云端构成的体系结构

传统云服务中,云服务器一直处在活跃状态,增加了远程云端的能源消耗。休眠机制是一种有效减少能源消耗的人为手段。借助于虚拟化技术,在同一台物理机上的虚拟机中引入周期性同步休眠机制,不同物理机上的虚拟机引入周期性异步休眠机制。在每个物理机上设置一个休眠定时器。当物理机上的全部任务结束服务时,该定时器触发,部署在该物理机上的全部虚拟机同时进入休眠状态。不同物理机上的定时器的触发相互独立,也就是说,一个物理机上的虚拟机的休眠或唤醒,与另一个物理机上虚拟机无关。

基于所提出的体系结构,远程云端上的虚拟机具有 3 种状态:活跃状态、空闲状态和休眠状态。虚拟机的状态转换过程如图 2 所示。

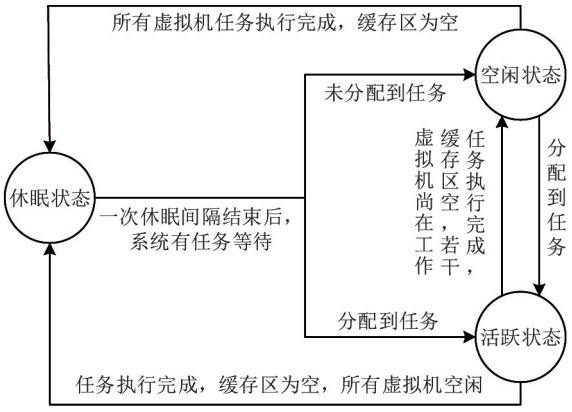


图 2 虚拟机的状态转移过程

（1）活跃状态。处于活跃状态的虚拟机为某一个任务提供服务。虚拟机服务完成当前任务后，按先来先服务规则为缓存区中等待的任务提供服务。若缓存区空，但其他虚拟机未全部空闲，则该虚拟机由活跃状态切换到空闲状态。若缓存区空且所有虚拟机空闲，则该虚拟机与其他虚拟机同步切换到休眠状态。

（2）空闲状态。处于空闲状态的虚拟机，可以随时为分配到的任务提供服务，也可以与其他虚拟机同步进入休眠状态。若分配到任务，空闲虚拟机可以立即切换到活跃状态。若所有虚拟机全部执行完成所分配的任务且缓存区空，该虚拟机与其他虚拟机同步切换到休眠状态。

（3）休眠状态。处于休眠状态的虚拟机暂时不再为任务提供服务。当虚拟机处于休眠状态时，新到达任务在系统缓存区中排队等待。当一个物理机的休眠定时器到期时，若系统缓存区空，重新启动休眠定时器，其上的虚拟机同时开始下一个休眠间隔，多个休眠间隔构成一个休眠期；若系统缓存区不空，全部虚拟机则同时结束休眠，依次处理系统缓存区滞留的任务。分配到任务的虚拟机由休眠状态切换到活跃状态，未被分配到任务的虚拟机由休眠状态切换到空闲状态。

1.2 系统模型

将移动设备视为连续工作的 M/M/1 排队，远程云端的每一个物理机视为虚拟机同步多重休假的 M/M/c_k 排队，建立系统模型。

令任务的产生服从参数为 $\lambda(0 < \lambda < +\infty)$ 的指数分布。假设一个物理机上的虚拟机服务能力相同，令物理机 $S_k(k = 1,2,\cdots,n)$ 上的虚拟机服务一个任务的时间服从参数为 $\mu_k(0 < \mu_k < +\infty)$ 的指数分布，令物理机休眠定时器长度服从参数为 $\theta_k(0 < \theta_k < +\infty)$ 的指数分布。

假设任务在本地执行的概率为 $q(0 \leqslant q \leqslant 1)$，任务卸载到远程云端的概率为 $1-q$。卸载到远程云端的任务将分配到某一个物理机上。假设某个任务分配到物理机 S_k 的概率为 ξ_k，则 $\xi_1 + \xi_2 + \cdots + \xi_n = 1$。基于以上假设，本地任务的到达服从参数为 $\lambda_0 = \lambda q$ 的指数分布，物理机 S_k 上的任务到达服从参数 $\lambda_k = \lambda(1-q)\xi_k$ 的指数分布。假设物理机具有无限容量的缓存。

本地移动设备的系统服务强度 ρ_0 定义为一个本地任务的服务时间内分配到本地移动设备的平均任务数。ρ_0 表示为

$$\rho_0 = \frac{\lambda_0}{\mu_0} \tag{1}$$

其中，μ_0 为本地移动设备的服务率。

云端物理机 S_k 的系统服务强度 ρ_k 定义为一个云端任务的服务时间内卸载到云端的平均任务数。ρ_k 表示为

$$\rho_k = \frac{\lambda_k}{c_k \mu_k} \quad k = 1,2,\cdots,n \tag{2}$$

为了使系统稳定，令本地移动设备的系统服务强度 ρ_0 和云端物理机 S_k 的系统服务强度 ρ_k 均小于 1。

令随机变量 $X_k(t) = i(i = 0,1,\cdots)$ 表示时刻 t 物理机 S_k 内的任务数。令随机变量 $Y_k(t) = j(j = 0,1)$ 表示时刻 t 物理机 S_k 上虚拟机所处的状态。当 $j = 0$ 时，表示虚拟机处在休眠状态；当 $j = 1$ 时，表示虚拟机处在活跃状态。$\{(X_k(t), Y_k(t)), t \geqslant 0\}$ 构成一个二维时间连续马尔科夫链，其状态空间表示为

$$\boldsymbol{\Omega}_k = \{(0,0)\} \cup \{(i,j): i \geqslant 1, j = 0,1\} \tag{3}$$

令 $\pi_k(i,j)$ 表示稳态下物理机 S_k 的系统水平为 i、系统阶段为 j 的概率分布。$\pi_k(i,j)$ 表示为

$$\pi_k(i,j) = \lim_{t\to\infty} P\{X_k(t) = i, Y_k(t) = j\} \quad i,j \in \boldsymbol{\Omega}_k \tag{4}$$

令 $\boldsymbol{\pi}_k(0) = \pi_k(0,0)$ 表示为稳态下系统水平为0的概率向量,令 $\boldsymbol{\pi}_k(i) = (\pi_k(i,0),\pi_k(i,1))$ 表示为稳态下系统水平为 i 的概率向量。二维连续时间马尔可夫链 $\{(X_k(t), Y_k(t)), t \geqslant 0\}$ 的稳态概率分布 $\boldsymbol{\Pi}_k$ 表示为

$$\boldsymbol{\Pi}_k = (\boldsymbol{\pi}_k(0), \boldsymbol{\pi}_k(1), \cdots) \tag{5}$$

2 系统模型的稳态分析

假设远程云端部署了 n 个异构物理机,物理机 S_k 上部署了 c_k 个虚拟机。物理机 S_k 可看作一个独立的同步多重休假 M/M/c_k 排队[8]。物理机 S_k 上虚拟机的状态转移机制如图3所示。

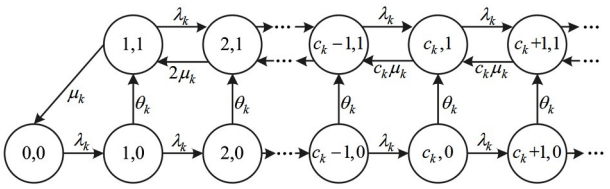


图3 物理机 S_k 上虚拟机的状态转移

由图3可以分析出一个物理机中多个虚拟机的状态转移过程。当 $j = 0$ 时,物理机 S_k 上 c_k 个虚拟机全部处于休眠状态。当 $j = 1$ 时,若系统内有 $i < c_k$ 个任务,则有 i 个虚拟机处于活跃状态, $c_k - i$ 个虚拟机处于空闲状态;若系统内有 $i \geqslant c_k$ 个任务,则物理机上部署的 c_k 个虚拟机全部处于活跃状态。

令 $\boldsymbol{Q}_k$ 表示二维连续时间马尔可夫链 $\{(X_k(t), Y_k(t)), t \geqslant 0\}$ 的一步转移率矩阵, $\boldsymbol{Q}_k(x, y)$ 表示经过一步转移后,系统水平从 $x(x = 0,1,\cdots)$ 到 $y(y = 0,1,\cdots)$ 的转移率子阵。为了表述方便,将 $\boldsymbol{Q}_k(x, x-1)$、$\boldsymbol{Q}_k(x, x)$ 与 $\boldsymbol{Q}_k(x, x+1)$ 分别记为 $\boldsymbol{B}_k(x)$、$\boldsymbol{A}_k(x)$ 与 $\boldsymbol{C}_k(x)$。

(1) 当 $x = 0$ 时,虚拟机一定处于休眠状态,且缓存区为空。当无任务到达时,系统水平和系统阶段均保持不变, $\boldsymbol{A}_k(0)$ 退化为一个数值, $\boldsymbol{A}_k(0) = -\lambda_k$。若有一个任务到达,系统水平由 $x = 0$ 变为 $x = 1$, 系统阶段不变,转移率为 λ_k,$\boldsymbol{C}_k(0)$ 为 1×2 维矩阵,表示为

$$\boldsymbol{C}_k(0) = (\lambda_k \quad 0) \tag{6}$$

(2) 当 $x = 1$ 时,虚拟机既可能处于休眠状态,也可能处于活跃状态。

首先,讨论系统水平下降的情形。当虚拟机处于休眠状态时,任务不可能产生离去。当虚拟机处于活跃状态时,若处理完成一个任务,系统水平由 $x = 1$ 变为 $x = 0$, 虚拟机由活跃状态切换到休眠状态,系统阶段由 $j = 1$ 变为 $j = 0$, 转移率为 μ_k。$\boldsymbol{B}_k(1)$ 为 2×1 维矩阵,表示为

$$\boldsymbol{B}_k(1) = \begin{pmatrix} 0 \\ \mu_k \end{pmatrix} \tag{7}$$

其次,讨论系统水平不变的情形。当虚拟机处于休眠状态时,若无任务到达且休眠定时器未到期,转移率为 $-(\lambda_k + \theta_k)$; 若休眠定时器到期,虚拟机由休眠状态切换到活跃状态,系统阶段由 $j = 0$ 变为 $j = 1$, 转移率为 θ_k。当虚拟机处于活跃状态时,若无任务到达且虚拟机未处理完成当前任务,转移率为 $-(\lambda_k + \mu_k)$, 此时,虚拟机无法进入休眠状态。$\boldsymbol{A}_k(1)$ 为 2×2 维矩阵,表示为

$$\boldsymbol{A}_k(1) = \begin{pmatrix} -(\lambda_k + \theta_k) & \theta_k \\ 0 & -(\lambda_k + \mu_k) \end{pmatrix} \tag{8}$$

最后,讨论系统水平增加的情形。无论虚拟机处于休眠状态还是活跃状态,若有一个任务到达,系统水平由 $x = 1$ 变为 $x = 2$, 系统阶段不变,转移率为 λ_k。$\boldsymbol{C}_k(1)$ 为 2×2 维矩阵,表示为

$$\boldsymbol{C}_k(1) = \begin{pmatrix} \lambda_k & 0 \\ 0 & \lambda_k \end{pmatrix} \tag{9}$$

(3) 当 $1 < x \leqslant c_k$ 时,虚拟机同样可能处于休眠状态或活跃状态。

首先,讨论系统水平下降的情形。当虚拟机处于休眠状态时,任务不可能产生离去。当虚拟机处于活跃状态时,若处理完成一个任务,系统水平由 x 变为 $x-1$, 转移率为 $x\mu_k$, 此时,物理机 S_k 还存在未处理完成的任务,虚拟机无法进入休眠状态。$\boldsymbol{B}_k(x)$ 为 2×2 维矩阵,表示为

$$\boldsymbol{B}_k(x) = \begin{pmatrix} 0 & 0 \\ 0 & x\mu_k \end{pmatrix} \tag{10}$$

其次,讨论系统水平不变的情形。当虚拟机处于休眠状态时,若无任务到达且休眠定时器未到期,转移率为 $-(\lambda_k + \theta_k)$; 若休眠定时器到期,虚拟机由休眠状态切换到活跃状态,系统阶段由 $j = 0$ 变为

$j=1$，转移率为 θ_k。当虚拟机处于活跃状态时，若无任务到达且虚拟机未处理完成当前任务，转移率为 $-(\lambda_k+x\mu_k)$，此时，虚拟机无法进入休眠状态。$\boldsymbol{A}_k(x)$ 为 2×2 维矩阵，表示为

$$\boldsymbol{A}_k(x)=\begin{pmatrix}-(\lambda_k+\theta_k) & \theta_k\\ 0 & -(\lambda_k+x\mu_k)\end{pmatrix} \tag{11}$$

最后，讨论系统水平增加的情形。无论虚拟机处于休眠状态还是活跃状态，若有一个任务到达，系统水平由 x 变为 $x+1$，系统阶段不变，转移率为 λ_k。$\boldsymbol{C}_k(x)$ 为 2×2 维矩阵，表示为

$$\boldsymbol{C}_k(x)=\begin{pmatrix}\lambda_k & 0\\ 0 & \lambda_k\end{pmatrix} \tag{12}$$

（4）当 $x>c_k$ 时，虚拟机仍然可能处于休眠状态或者活跃状态。

首先，讨论系统水平下降的情形。当虚拟机处于休眠状态时，任务不可能产生离去。当虚拟机处于活跃状态时，有 c_k 个任务同时接受服务，若处理完成一个任务，系统水平由 x 变为 $x-1$，转移率为 $c_k\mu_k$，此时，虚拟机同样无法进入休眠状态。$\boldsymbol{B}_k(x)$ 为 2×2 维矩阵，表示为

$$\boldsymbol{B}_k(x)=\begin{pmatrix}0 & 0\\ 0 & c_k\mu_k\end{pmatrix} \tag{13}$$

其次，讨论系统水平不变的情形。当虚拟机处于休眠状态时，若无任务到达且休眠定时器未到期，转移率为 $-(\lambda_k+\theta_k)$；若休眠定时器到期，虚拟机由休眠状态切换到活跃状态，系统阶段由 $j=0$ 变为 $j=1$，转移率为 θ_k。当虚拟机处于活跃状态时，若无任务到达且虚拟机未处理完成当前任务，转移率为 $-(\lambda_k+c_k\mu_k)$，此时，虚拟机无法进入休眠状态。$\boldsymbol{A}_k(x)$ 为 2×2 维矩阵，表示为

$$\boldsymbol{A}_k(x)=\begin{pmatrix}-(\lambda_k+\theta_k) & \theta_k\\ 0 & -(\lambda_k+c_k\mu_k)\end{pmatrix} \tag{14}$$

最后，讨论系统水平增加的情形。无论虚拟机处于休眠状态还是活跃状态，若有一个任务到达，系统水平由 x 变为 $x+1$，系统阶段不变，转移率为 λ_k。$\boldsymbol{C}_k(x)$ 为 2×2 维矩阵，表示为

$$\boldsymbol{C}_k(x)=\begin{pmatrix}\lambda_k & 0\\ 0 & \lambda_k\end{pmatrix} \tag{15}$$

由上述分析可得，转移率子阵 $\boldsymbol{A}_k(x)$ 与 $\boldsymbol{B}_k(x)$ 均从系统水平 c_k 开始重复，$\boldsymbol{C}_k(x)$ 从 1 开始重复。将重复的 $\boldsymbol{A}_k(x)$ 与 $\boldsymbol{B}_k(x)$ 分别用 $\boldsymbol{A}_k$ 与 $\boldsymbol{B}_k$ 来表示，将重复的 $\boldsymbol{C}_k(x)$ 用 $\boldsymbol{C}_k$ 来表示，则马尔可夫链 $\{(X_k(t),Y_k(t)),t\geqslant0\}$ 的一步转移率矩阵 $\boldsymbol{Q}_k$ 可以表示为如下分块形式。

$$\boldsymbol{Q}_k=\begin{pmatrix}\boldsymbol{A}_k(0) & \boldsymbol{C}_k(0) & & & & & \\ \boldsymbol{B}_k(1) & \boldsymbol{A}_k(1) & \boldsymbol{C}_k & & & & \\ & \ddots & \ddots & \ddots & & & \\ & & \boldsymbol{B}_k(c_k-1) & \boldsymbol{A}_k(c_k-1) & \boldsymbol{C}_k & & \\ & & & \boldsymbol{B}_k & \boldsymbol{A}_k & \boldsymbol{C}_k & \\ & & & & \ddots & \ddots & \ddots\end{pmatrix} \tag{16}$$

从一步转移率矩阵 $\boldsymbol{Q}_k$ 的结构可知，状态的转移只发生在相邻的系统水平之间。因此，二维连续时间马尔可夫链 $\{(X_k(t),Y_k(t)),t\geqslant0\}$ 可以看成一种拟生灭过程。该过程正常返的充分必要条件是矩阵方程

$$\boldsymbol{R}_k^2\boldsymbol{B}_k+\boldsymbol{R}_k\boldsymbol{A}_k+\boldsymbol{C}_k=0 \tag{17}$$

的最小非负解 $\boldsymbol{R}_k$（也称为率阵）的谱半径 $\mathrm{Sp}(\boldsymbol{R}_k)<1$。

由 $\boldsymbol{Q}_k$ 子阵结构，可设率阵 $\boldsymbol{R}_k=\begin{pmatrix}r_k^{11} & r_k^{12}\\ 0 & r_k^{22}\end{pmatrix}$，将 $\boldsymbol{A}_k$、$\boldsymbol{B}_k$、$\boldsymbol{C}_k$ 与 $\boldsymbol{R}_k$ 代入式(17)，可得率阵 $\boldsymbol{R}_k$ 如下。

$$\boldsymbol{R}_k=\begin{pmatrix}\dfrac{\lambda_k}{\lambda_k+\theta_k} & \rho_k\\ 0 & \rho_k\end{pmatrix} \tag{18}$$

根据率阵 $\boldsymbol{R}_k$ 的解析结果，可以看出率阵 $\boldsymbol{R}_k$ 的谱半径 $\mathrm{Sp}(\boldsymbol{R}_k)=\{r_k^{11},r_k^{22}\}<1$，所以二维马尔可夫链 $\{(X_k(t),Y_k(t)),t\geqslant0\}$ 正常返。利用所求得的率阵 $\boldsymbol{R}_k$，构造方阵 $B[\boldsymbol{R}_k]$ 如下。

$$B[\boldsymbol{R}_k]=\begin{pmatrix}\boldsymbol{A}_k(0) & \boldsymbol{C}_k(0) & & & & \\ \boldsymbol{B}_k(1) & \boldsymbol{A}_k(1) & \boldsymbol{C}_k & & & \\ & \ddots & \ddots & \ddots & & \\ & & \boldsymbol{B}_k(c_k-1) & \boldsymbol{A}_k(c_k-1) & \boldsymbol{C}_k & \\ & & & \boldsymbol{B}_k & \boldsymbol{R}_k\boldsymbol{B}_k+\boldsymbol{A}_k & \end{pmatrix} \tag{19}$$

由平衡方程及归一化条件可得如下方程组:

$$\begin{cases}(\boldsymbol{\pi}_k(0),\boldsymbol{\pi}_k(1),\cdots,\boldsymbol{\pi}_k(c_k))B[\boldsymbol{R}_k] = \boldsymbol{0}\\(\boldsymbol{\pi}_k(0),\cdots,\boldsymbol{\pi}_k(c_k-1))\boldsymbol{e}+\boldsymbol{\pi}_k(c_k)(\boldsymbol{I}-\boldsymbol{R}_k)^{-1}\boldsymbol{e}=1\end{cases} \tag{20}$$

其中,$\boldsymbol{I}$ 为单位矩阵,$\boldsymbol{e}$ 为全 1 列向量。

系统的稳态分布表示为

$$\pi_k(i,0) = \pi_k(0,0)\left(\frac{\lambda_k}{\lambda_k+\theta_k}\right)^i \quad i \geqslant 0 \tag{21}$$

$$\pi_k(i,1) = \begin{cases}\pi_k(0,0)\dfrac{1}{i!}\left(\dfrac{\lambda_k}{\mu_k}\right)^i\alpha_k(i) & 1 \leqslant i \leqslant c_k-1\\ \pi_k(c_k-1,1)\rho_k^{i-c_k+1}+\beta_k(i) & i \geqslant c_k\end{cases} \tag{22}$$

其中,

$$\alpha_k(i) = \sum_{l=0}^{i-1} l!\left(\frac{u_k}{\lambda_k+\theta_k}\right)^l \quad 1 \leqslant i \leqslant c_k-1$$

$$\beta_k(i) = \rho_k\pi_k(c_k-1,0)\sum_{j=0}^{i-c_k}\rho_k^j\left(\frac{\lambda_k}{\lambda_k+\theta_k}\right)^{i-c_k-j} \quad i \geqslant c_k$$

$\pi_k(0,0)$ 由归一化条件给出如下。

$$\begin{aligned}\pi_k(0,0) = \Bigg(&\sum_{i=1}^{c_k-1}\frac{1}{i!}\left(\frac{\lambda_k}{\mu_k}\right)^i\alpha_k(i)\\&+\frac{\rho_k}{1-\rho_k}\frac{1}{(c_k-1)!}\left(\frac{\lambda_k}{\mu_k}\right)^{c_k-1}\alpha_k(c_k-1)\\&+\frac{\lambda_k+\theta_k}{\theta_k}\left(1+\frac{\rho_k}{1-\rho_k}\left(\frac{\lambda_k}{\lambda_k+\theta_k}\right)^{c_k-1}\right)\Bigg)^{-1}\end{aligned} \tag{23}$$

3 系统性能指标

任务响应时间定义为从任务的产生时刻开始,到任务结束服务离开系统为止所经历的时间段。

分配到本地执行的任务平均响应时间 W_0 包括任务在本地缓存区的平均等待时间与在本地处理器的平均服务时间。根据排队模型 M/M/1 的解析结果,给出本地任务平均响应时间 W_0 的表达式为

$$W_0 = \frac{1}{\mu_0-\lambda_0} \tag{24}$$

卸载到远程云端的任务平均响应时间包括任务在缓存区的平均等待时间与在虚拟机上的平均服务时间。由第 2 节给出的系统模型的稳态分析结果计算出平均等待队长的表达式,进一步利用 Little 公式[9]得出物理机 S_k 上任务平均响应时间 W_k 的表达式为

$$W_k = \frac{1}{\lambda_k}\left(\sum_{i=c_k}^{\infty}(i-c_k)(\pi_k(i,0)+\pi_k(i,1))\right)+\frac{1}{\mu_k} \tag{25}$$

任务在本地执行的概率为 $q(0 \leqslant q \leqslant 1)$,卸载到远程云端的概率为 $1-q$,卸载到远程云端的任务分配给物理机 S_k 的概率为 ξ_k。综合式(24)给出的本地任务平均响应时间 W_0 和式(25)给出的物理机 S_k 任务平均响应时间 W_k,得出任务平均响应时间 W 为

$$W = qW_0+\sum_{k=1}^{n}(1-q)\xi_kW_k \tag{26}$$

移动设备处于活跃状态时,其处理器高速运转,令其运行功率表示为 P_0^{Active}。移动设备处于空闲状态时,其处理器低速运转,令其运行功率表示为 P_0^{Idle}。显然,$P_0^{Active} > P_0^{Idle}$。移动设备平均运行功率 P_0 为

$$P_0 = \rho_0P_0^{Active}+(1-\rho_0)P_0^{Idle} \tag{27}$$

其中,ρ_0 表示为移动设备处于活跃状态的概率。

虚拟机处于活跃状态时,其处理器高速运转,令 P_k^{Active} 表示物理机 S_k 支撑一个活跃虚拟机的运行功率。虚拟机处于空闲状态时,其处理器低速运转,令 P_k^{Idle} 表示物理机 S_k 支撑一个空闲虚拟机的运行功率。虚拟机处于休眠状态时,其处理器超低速运转,令 P_k^{Sleep} 表示物理机 S_k 支撑一个休眠虚拟机的运行功率。显然,$P_k^{Active} > P_k^{Idle} > P_k^{Sleep}$。物理机 S_k 平均运行功率 P_k 表达式为

$$\begin{aligned}P_k = &\left(\sum_{i=1}^{c_k-1}i\pi_k(i,1)+\sum_{i=c_k}^{\infty}c_k\pi_k(i,1)\right)P_k^{Active}\\&+\sum_{i=1}^{c_k-1}(c_k-i)\pi_k(i,1)P_k^{Idle}+\sum_{i=0}^{\infty}c_k\pi_k(i,0)P_k^{Sleep}\end{aligned} \tag{28}$$

综合式(27)给出的移动设备平均运行功率 P_0 和式(28)给出的物理机 S_k 平均运行功率 P_k,得出系统平均运行功率 P 的表达式为

$$P = qP_0+\sum_{k=1}^{n}(1-q)\xi_kP_k \tag{29}$$

4 系统实验

为量化任务到达率 λ、休眠参数 θ_k、任务分配到移动设备本地的概率 q 及任务卸载到远程云端物理机 S_k 上的概率 ξ_k 对云系统任务调度策略的影响，进行系统实验，刻画以分钟(min)为单位的任务平均响应时间与以毫瓦(mW)为单位的系统平均运行功率的变化趋势。系统实验所用计算机的处理器为 Intel(R) Core(TM) i7-4790，运行频率为 3.60 GHz，内存为 8 GB。系统实验与系统优化在 Matlab R2016a 的环境下实现。

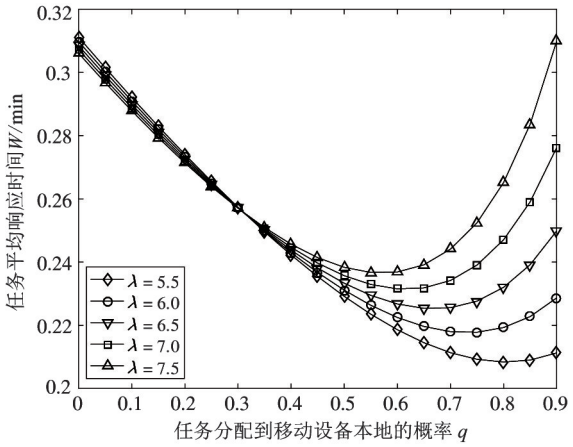
在保证系统稳定，即本地服务强度 $\rho_0 < 1$ 与远程云端物理机的服务强度 $\mathrm{Max}(\rho_1, \rho_2, \cdots, \rho_n) < 1$ 的约束下，设置如表 1 所示的系统实验参数。

表 1 系统实验参数设置

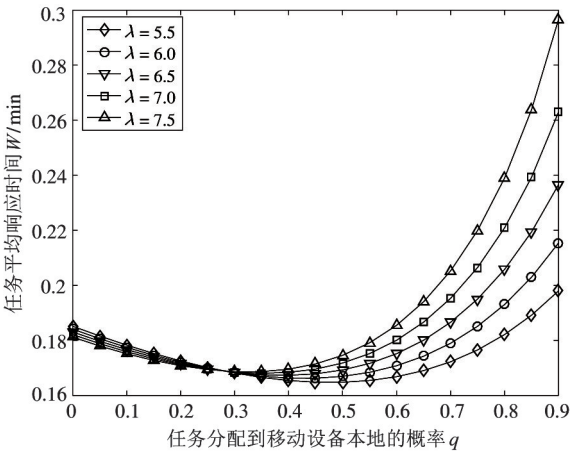
参数	设定值
本地处理器的服务速率 μ_0	10 个/min
物理机 S_1 上虚拟机的服务速率 μ_1	18 个/min
物理机 S_2 上虚拟机的服务速率 μ_2	15 个/min
物理机 S_3 上虚拟机的服务速率 μ_3	30 个/min
物理机 S_1 上部署虚拟机的数量 c_1	3 个
物理机 S_2 上部署虚拟机的数量 c_2	3 个
物理机 S_3 上部署虚拟机的数量 c_3	4 个
本地处理器处于活跃状态下的运行功率 P_0^{Active}	60 mW
本地处理器处于空闲状态下的运行功率 P_0^{Idle}	22 mW
物理机 S_1 支撑一个活跃虚拟机的运行功率 P_1^{Active}	45 mW
物理机 S_1 支撑一个空闲虚拟机的运行功率 P_1^{Idle}	20 mW
物理机 S_1 支撑一个休眠虚拟机的运行功率 P_1^{Sleep}	10 mW
物理机 S_2 支撑一个活跃虚拟机的运行功率 P_2^{Active}	40 mW
物理机 S_2 支撑一个空闲虚拟机的运行功率 P_2^{Idle}	20 mW
物理机 S_2 支撑一个休眠虚拟机的运行功率 P_2^{Sleep}	10 mW
物理机 S_3 支撑一个活跃虚拟机的运行功率 P_3^{Active}	50 mW
物理机 S_3 支撑一个空闲虚拟机的运行功率 P_3^{Idle}	20 mW
物理机 S_3 支撑一个休眠虚拟机的运行功率 P_3^{Sleep}	10 mW

使用表 1 设定的参数，固定卸载到远程云端物理机 S_1、S_2、S_3 的概率 ($\xi_1 = 0.2225$、$\xi_2 = 0.3432$、$\xi_3 = 0.4343$)，考虑不同的休眠参数 $\theta_k(k = 1,2,3)$ 进行系统实验，图 4 刻画了任务到达率 λ 与任务分配到移动设备本地的概率 q 对任务平均响应时间 W 的影响。

横向观察图 4，随着任务分配到移动设备本地的概率 q 的增大，任务平均响应时间 W 先呈现下降趋势，在到达最低点之后开始上升。当任务分配到移动设备本地的概率较小时，任务在远程云端物理机的响应时间占平均响应时间的主导地位。随着任务分配到移动设备本地的概率的增大，任务在远程云端物理机缓存区的平均等待时间随之减少，任务平均响应时间降低。当任务分配到移动设备本地的概率较大时，任务在移动设备的响应时间占平均响应时间的主导地位。随着任务分配到移动设备本地的概率的增大，任务在移动设备本地的平均等待时间随之增大，任务平均响应时间上升。



(a) $\theta_1 = \theta_2 = \theta_3 = 3$



(b) $\theta_1 = \theta_2 = \theta_3 = 5$

图 4 任务平均响应时间的变化趋势

纵向观察图 4，随着任务到达率 λ 的增大，任务平均响应时间 W 的变化趋势与任务分配到移动设

备本地的概率 q 有关。当任务分配到移动设备本地的概率较小时,更多的任务卸载到远程云端物理机接受服务。任务到达率越大,虚拟机进入休眠状态的概率越小,任务平均响应时间随之减少。当任务分配到移动设备本地的概率较大时,分配到移动设备本地的任务越多,任务在移动设备本地的等待时间变长,任务平均响应时间随之增大。

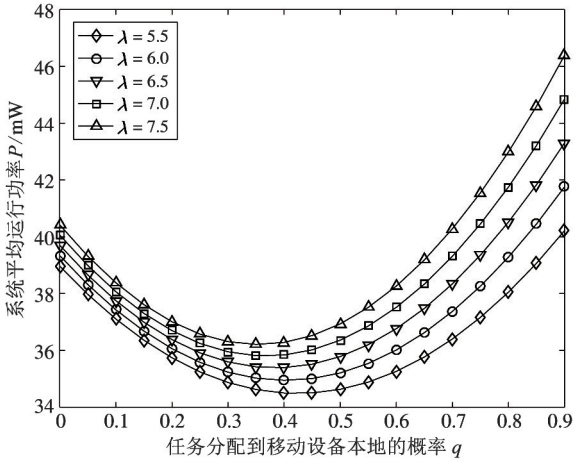
对比观察图 4(a)和图 4(b),随着休眠参数 $\theta_k(k=1,2,3)$ 的增大,任务平均响应时间 W 呈现下降趋势。任务分配到本地设备的概率较小时,更多的任务卸载到远程云端物理机。当休眠参数较小时,任务在缓存区中等待虚拟机结束休眠的时间较长,导致任务平均响应时间加大。这种情况下,本文所提策略中的休眠机制对任务平均响应时间的影响较大。而当休眠参数较大时,任务在缓存区中等待虚拟机结束休眠的时间相对缩短,任务平均响应时间随之减少。这种情况下,本文所提策略中的休眠机制对任务平均响应时间的影响较小。

图 5 刻画了任务到达率 λ 与任务分配到移动设备本地的概率 q 对系统平均运行功率 P 的影响。

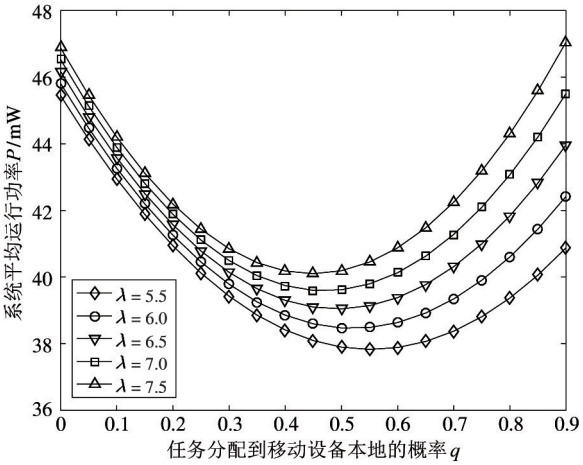
横向观察图 5,随着任务分配到移动设备本地的概率 q 的增大,系统平均运行功率 P 先呈现下降趋势,在到达最低点之后开始逐渐上升。当任务分配到移动设备本地的概率较小时,更多的任务卸载到远程云端物理机接受服务,远程云端物理机的运行功率占系统平均运行功率 P 的主导地位。随着任务分配到移动设备本地的概率增大,卸载到远程云端物理机的任务量减少,物理机上的虚拟机处于活跃状态的可能性减小,物理机的运行功率下降,系统平均运行功率下降。当任务分配到移动设备本地的概率较大时,更多的任务分配到移动设备本地接受服务,移动设备本地的运行功率占系统平均运行功率 P 的主导地位。随着任务分配到移动设备本地的概率增大,分配到移动设备本地的任务增多,移动设备功耗增大,系统平均运行功率上升。

纵向观察图 5,随着任务到达率 λ 的增大,系统平均运行功率 P 随之增大。任务到达率越大,意味着任务量越多,移动设备与物理机的功耗增大,系统平均运行功率上升。

对比观察图 5(a)和图 5(b),随着休眠参数 $\theta_k(k=1,2,3)$ 的增大,系统平均运行功率 P 呈现上升趋势。随着休眠参数的增大,虚拟机进入休眠状态的可能性降低,而虚拟机处于活跃状态或空闲状态的可能性增大,系统平均运行功率上升。



(a) $\theta_1 = \theta_2 = \theta_3 = 3$



(b) $\theta_1 = \theta_2 = \theta_3 = 5$

图 5　系统平均运行功率的变化趋势

比较图 4 与图 5 可以看出,最低的平均响应时间所对应的任务分配到移动设备本地的概率无法实现最低的能耗水平。实际应用中,需根据实际应用的业务特点设置任务分配到移动设备本地的概率。

5　系统优化

系统实验所揭示的任务平均响应时间和系统平均运行功率的变化规律表明,任务到达率、休眠参数

与任务分配到移动设备本地的概率是影响系统性能的重要因素。在不同的任务到达率下,合理设置任务分配到移动设备本地的概率能够实现响应性能与能耗水平的合理均衡。

对任务平均响应时间 W 与系统平均运行功率 P 进行归一化处理,构造无量纲系统成本函数如下。

$$F(q) = f_1 \times \frac{W - W_{\min}}{W_{\max} - W_{\min}} + f_2 \times \frac{P - P_{\min}}{P_{\max} - P_{\min}} \tag{30}$$

其中,f_1 表示任务平均响应时间对系统成本函数的影响因子,f_2 表示系统平均功率对系统成本函数的影响因子[10]。当任务分配到移动设备本地的概率 q 的取值在[0, 0.9]时,任务平均响应时间 W 的最大值和最小值表示为 $W_{\max}$ 和 $W_{\min}$,系统平均运行功率 P 的最大值和最小值表示为 $P_{\max}$ 和 $P_{\min}$。

为了实现系统成本函数的最小化,利用智能寻优算法找出最优任务分配到移动设备本地的概率。相比于一般的优化算法,鲸鱼优化算法的收敛速度较快,能够在较短的时间内寻找到目标函数的最优解。算法中的每一头鲸鱼所处的位置代表目标函数的一个可行解,采用随机和最佳搜索代理模拟鲸鱼的捕猎行为,并使用螺旋模型模拟座头鲸的汽泡网攻击机制[11]。本文中,增加了混沌初始化用来提高鲸鱼优化算法的全局搜索能力。算法的主要步骤如下所示。

步骤 1 初始化算法执行的最大迭代次数 $\mathrm{Max}_{iter}=10\,000$,鲸鱼种群规模 $N = 1000$;设置任务分配到移动设备本地的概率 q 的上限 $q_{\mathrm{up}}=1$ 及下限 $q_{\mathrm{low}} = 0$,当前迭代次数 $iter = 1$。

步骤 2 利用混沌方程初始化鲸鱼种群的位置。

$q_1(iter) = rand \times (q_{\mathrm{up}} - q_{\mathrm{low}}) + q_{\mathrm{low}}$

for $i = 2:N$

$q_i(iter) = \gamma \times q_{i-1}(iter)(1 - q_{i-1}(iter)) + 0.02$

% $q_i(iter)$ 代表当前迭代过程中第 $i(i = 1, 2,\cdots,N)$ 头鲸鱼的位置。$\gamma = 3.5$ 表示混沌初始化的混沌因子。

endfor

步骤 3 设第 1 条鲸鱼位置为当前最优位置 q^*,利用式(30)计算系统成本函数 $F(q^*)$。

步骤 4 利用式(30)计算全部鲸鱼所对应的系统成本函数 $F(q_i(iter))$, $i = 1,2,\cdots,N$, 找出当前最优鲸鱼位置 q^*。

for $i = 1:N$

if $F(q_i(iter)) < F(q^*)$

$q^* = q_i(iter)$

$F(q^*) = F(q_i(iter))$

endif

endfor

步骤 5 根据鲸鱼捕食行为更新鲸鱼位置。

for $i = 1:N$

$A = \left(2 - \frac{2 \times iter}{\mathrm{Max}_{iter}}\right)(2 \times rand - 1)$

$C = 2 \times rand$% A 和 C 为鲸鱼移动系数

$p = rand$

if $p < 0.5$

if $|A| >= 1$% 采用随机代理搜索方式搜索猎物位置

$Z = floor(rand \times N + 1)$

$D = |C \times q_Z(iter) - q_i(iter)|$

$q_i(iter + 1) = q_Z(iter) - A \times D$

else% 采用最佳代理搜索方式搜索猎物位置

$D = |C \times q^* - q_i(iter)|$

$q_i(iter + 1) = q^* - A \times D$

endif

else% 采用螺旋模型方式搜索猎物位置

$l = rand \times \left(-2 + \frac{iter}{\mathrm{Max}_{iter}}\right) + 1$

$D = |q^* - q_i(iter)|$

$q_i(iter + 1) = q^* + D \times e^l \times \cos(2 \times \pi \times l)$

endif

endfor

步骤 6 判断迭代过程是否完成。

if $iter < \mathrm{Max}_{iter}$

$iter = iter + 1$

算法跳转到步骤 4

endif

步骤 7 输出最优鲸鱼位置为最优分配到移动设备本地的概率 q^*,给出最小系统成本函 $F(q^*)$。

设置系统成本函数的影响因子 $f_1 = 2.0$、$f_2 = 1.6$,利用改进的鲸鱼优化算法给出不同任务到达率下任务分配到移动设备本地的最优概率,优化结果如表 2 所示。

表 2 任务分配到移动设备本地的概率的优化结果

任务到达率 λ	任务分配到本地的最优概率 q^*	最小系统成本 $F(q^*)$
5.5	0.3526	0.0134
6.0	0.3325	0.0577
6.5	0.3144	0.0992
7.0	0.3001	0.1387
7.5	0.2851	0.1763

从表 2 的数值结果可以看出,任务分配到移动设备本地的最优概率随着任务到达率的增大逐渐减小。移动设备的本地处理能力有限,随着任务到达率的增大,移动设备本地的任务量负载加重,因此,任务分配到本地执行的最优概率呈下降趋势。

6 结 论

综合考虑云用户的响应性能与云系统的能耗水平,在远程云端引入周期性休眠机制,提出了一种云计算任务调度策略。基于远程云端,考虑异构物理机,建立同步多重休假 M/M/c_k 排队模型,给出了系统模型的稳态分布。基于 Matlab 进行了系统实验,实验结果表明,任务平均响应时间与系统平均运行功率之间存在折衷关系。设定任务平均响应时间和系统平均运行功率的影响因子,构建了系统成本函数。利用混沌方程初始化种群位置,改进鲸鱼优化算法,给出了任务分配到移动设备本地的概率的优化结果,实现了系统成本函数的最小化。

参考文献

[1] Jadad H, Touzene A, Day K, et a1. Context-aware prediction model for offloading mobile application tasks to mobile cloud environments[J]. *International Journal of Cloud Applications and Computing*, 2019, 9(3): 751-774

[2] Jayanetti A, Buyya R. A joint host and network optimization algorithm for energy-efficient workflow scheduling in cloud data centers[C]//IEEE International Conference on Utility and Cloud Computing, Auckland, New Zealand, 2019: 199-208

[3] Vakilinia S, Heidarpour B, Cheriet M. Energy efficient resource allocation in cloud computing environments[J]. *IEEE Access*, 2017, 4(99): 8544-8557

[4] Duan L D, Zhan D Y, Hohnerlein J, et al. Optimizing cloud data center energy efficiency via dynamic prediction of CPU idle intervals[C]//IEEE International Conference on Cloud Computing, New York, USA, 2015: 985-988

[5] 王晓琛, 王宇廷, 张丽媛, 等. 基于(N, T)休眠机制的云计算中心节能策略及优化[J]. 高技术通讯, 2020, 30(8): 805-813

[6] 程春玲, 王颖, 张登银. 云计算中基于动态阈值的服务器唤醒策略[J]. 系统工程与电子技术, 2015, 37(6): 1437-1445

[7] Cao J W, Li K Q. Optimal power allocation and load distribution for multiple heterogeneous multicore server processors across clouds and data centers[J]. *IEEE Transactions on Computers*, 2014, 63(1): 45-58

[8] Li K. Improving multicore server performance and reducing energy consumption by workload dependent dynamic power management[J]. *IEEE Transactions on Cloud Computing*, 2016, 4(2): 122-137

[9] 孙健, 丁日佳, 陈艳艳. M/M/c 型与 M/M/1 型排队系统对比仿真[J]. 北京工业大学学报, 2016, 42(9): 1324-1331

[10] 王秀双, 金顺福. 基于新型休眠机制的云任务调度策略的研究[J]. 高技术通讯, 2018, 28(11-12): 907-914

[11] Mirjalili S, Lewis A. The whale optimization algorithm[J]. *Advances in Engineering Software Telecommunications Technologies*, 2016, 95(5): 51-67

Task scheduling strategy in cloud computing and its performance optimization based on heterogeneous physical machines

Fan Baozhi, Wang Kaiyu, Bai Xiaojun, Jin Shunfu
(School of Information Science and Engineering, Yanshan University, Qinhuangdao 066004)

Abstract

To reduce the energy consumption of the cloud system under the premise of ensuring the response performance of the cloud users, a task scheduling strategy is proposed. In the proposed strategy, the local processor of the mobile device continues to work, the virtual machines in a physical machine sleep synchronously, and the virtual machines in different physical machine sleep asynchronously. For the heterogeneous physical machines in cloud computing, a queueing model with a synchronous multiple vacation is established. By using the quasi birth-death process and the matrix geometric solution, the steady-state distribution of the queueing model is given, and the expressions of the average response time of the tasks and the average power of the system are derived. The experimental results show that there is a trade-off between different performance measures when setting the assigning probability of the tasks to the local processor. The traditional whale optimization algorithm is improved by introducing Logistic mapping chaos mechanism. The task scheduling strategy is optimized with the minimum system cost.

Key words: cloud computing, task scheduling strategy, synchronous multiple vacation, average response time, average power