高技术通讯 2021 年 第31 卷 第12 期:1269-1279

doi:10.3772/j.issn.1002-0470.2021.12.005

基于半休眠模式的云计算任务卸载策略及性能研究①

宋 家② 余 靖③ 金顺福

(燕山大学信息科学与工程学院 秦皇岛 066004)

摘 要 云系统中的任务卸载是一种将密集的移动应用程序迁移到云端计算的机制。考虑云用户服务质量和云系统能耗水平,在云端引入周期性同步半休眠模式,提出一种云计算任务卸载策略。针对不同负载情况下的虚拟机服务速率,构建多服务台同步多重工作休假排队模型。基于拟生灭过程和矩阵几何解方法,得到云用户的平均响应时间与云系统的能源消耗。进行系统实验,评估卸载策略的系统性能,并验证其有效性。基于权重分配构建系统成本函数,改进海鸥优化算法,给出云计算任务卸载策略的智能优化方案,实现系统成本的最小化。

关键词 云计算;任务卸载;半休眠;工作休假排队;智能优化

0 引 言

随着智能手机的大量普及和移动通信网络的迅猛发展,各类网络应用程序,如交互式游戏、虚拟现实、自然语言处理等层出不穷。这些应用程序通常是计算密集型和高耗能的,不适合在计算能力和电池供应有限的移动设备上工作。为了满足复杂应用程序的资源需求,移动云计算的概念应运而生。如何设计同时保证云用户的服务质量和云服务供应商运营利润的云计算卸载策略成为热门研究问题[1-2]。

过去的几年中,云数据中心的能源消耗持续增长,云数据中心消耗的大量能量对环境造成了不利影响。文献[3]提出了以节省能量消耗为目标的云资源调度优化管理程序,利用虚拟机监控程序分配资源,实现云计算资源分配的最小能耗和最高效率。文献[4]为了解决云用户的服务质量和能耗等问题提出了一种云计算环境下动态资源分配模型,使用间距多目标蚁群算法将云系统状态的动态变化与任务调度相结合,通过调整未使用的虚拟机,节省云系统的能源消耗。上述文献均采用资源优化配置的方式节省云系统能源消耗。

考虑到大部分云资源的使用率都较低,部分研究通过应用休眠机制以实现降低能源消耗的目的。文献[5]为提高云系统节能水平,同时满足云用户响应性能,引入休眠延迟机制与休眠唤醒阈值,提出了一种云资源调度策略。建立带有休假延迟与启动过程的同步 N-策略多重休假排队模型,评估了所提策略的系统性能。文献[6]以实现绿色云计算为目的,将基于唤醒阈值的休眠模式引入到集群虚拟机中,提出了一种云任务分配策略。建立异步 N-策略多重休假排队模型,给出了任务平均延迟和系统节能率的表达式。在以上的休眠机制中,处于休眠状态的虚拟机完全不工作,虽然有效降低了云系统能源消耗,但是云用户的服务质量得不到保证。

本文兼顾云用户的服务质量与云系统的能源消耗水平,在云端引入周期性同步半休眠模式,提出一种新型的云计算任务卸载策略。考虑不同负载情况下的云端虚拟机服务速率,构建连续时间工作休假的排队模型,研究云用户的随机行为。结合系统中任务请求的数量、本地处理器状态以及云端虚拟机

① 国家自然科学基金(61872311, 61973261)资助项目。
② 女,1996 年生,硕士生;研究方向:计算机网络性能分析;E-mail: ysusj750626@163.com。
③ 通信作者,E-mail: xyyj@ysu.edu.cn。
(收稿日期:2020-11-24)

状态,建立二维连续时间的 Markov 链,进行云用户平均响应时间与云系统能源消耗的理论分析。考虑不同休眠参数和不同系统负载,基于 Matlab 进行系统实验,给出系统性能指标的统计结果。构造系统成本函数,利用 Logistic 映射混沌方法初始化种群位置,改进海鸥优化算法。以最小化系统成本函数为目标,优化任务请求本地分配概率,为云计算任务卸载策略的实施提供理论依据。

1　云计算任务卸载策略及系统模型

1.1　云计算任务卸载策略

针对移动设备存储空间不足且处理能力有限等问题,将部分任务请求迁移到云端[7]。云服务通过分布式软件依托于云计算环境。在云数据服务中心上通常部署一个或多个物理机,而一个物理机上又可以部署多个虚拟机。当一个物理机上没有需要被执行的任务时,如果其上部署的虚拟机保持空闲状态,会因此产生大量的能耗,让空闲的虚拟机进入休眠状态是降低云计算能耗的有效方法之一。然而,休眠模式也可能会降低响应性能。因此在云端引入半休眠模式,提出一种新的云计算任务卸载策略。

产生于移动设备的任务请求首先汇聚于接入点,然后在本地负载均衡器的调度下就近分配到本地处理器接受服务,或卸载至云端接受服务。卸载到云端的任务请求由云端负载均衡器分配给某一个物理机。云计算任务卸载策略的系统架构如图 1 所示。

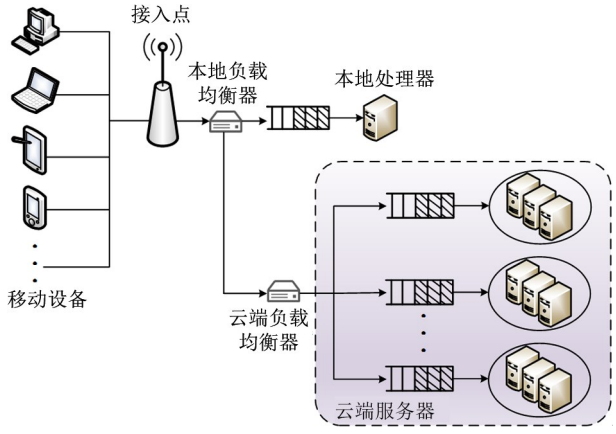


图 1　云计算任务卸载策略系统架构

当一个物理机上的任务请求全部完成时,若缓存区中也无等待的任务请求,则该物理机上部署的全部虚拟机同时由常规状态进入半休眠状态。在常规状态下的虚拟机有高速工作状态和高速空闲状态两种状态。在半休眠状态下,虚拟机不是完全停止工作,而是以低速率运转,此时包含有低速工作状态和低速空闲状态。在所提出的云计算任务卸载策略中,虚拟机共有 4 种状态:高速工作状态、高速空闲状态、低速工作状态和低速空闲状态。虚拟机的状态转换过程如图 2 所示。

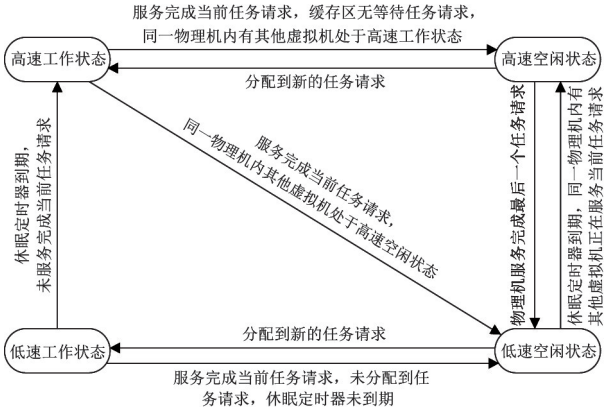


图 2　虚拟机的状态转换图

（1）高速工作状态。物理机及物理机上部署的全部虚拟机正常工作,此状态下物理机中至少有一个虚拟机正在服务任务请求。高速工作状态下到达缓存区的任务请求依次在虚拟机接受服务。当服务完成最后一个任务请求时,物理机启动休眠定时器,开始一段半休眠期,物理机上的所有虚拟机同时进入低速空闲状态。若处于高速工作状态的虚拟机服务完成当前的任务请求,未分配到新的任务请求,且同一物理机内有其他虚拟机还处于高速工作状态,则该虚拟机转换到高速空闲状态。

（2）高速空闲状态。处于高速空闲状态的虚拟机若分配到新的任务请求,则立即转换到高速工作状态提供服务。若物理机服务完成最后一个任务请求,则启动休眠定时器,其上的所有虚拟机同时进入低速空闲状态。

（3）低速空闲状态。处于低速空闲状态的虚拟机可以随时为新到达的任务请求提供服务,不受半休眠定时器的控制。如果在休眠定时器所规定的范

围内没有任务请求到达,物理机将重新启动休眠定时器,开始一次新的半休眠期,其上的虚拟机均保持在低速空闲状态。当一个虚拟机处于低速空闲状态时,若休眠定时器到期,且同一物理机内有其他虚拟机未服务完成任务请求,则物理机结束半休眠,该虚拟机转换到高速空闲状态;若分配到新的任务请求,该虚拟机立即由低速空闲状态转换到低速工作状态。

(4) 低速工作状态。当一个虚拟机以较低的速率完成当前的任务请求时,如果缓存区没有其他等待的任务请求,该虚拟机将由低速工作状态转换到低速空闲状态;否则,该虚拟机保持在低速工作状态,以较低的服务速率为缓存区中等待的任务请求提供服务。当休眠定时器到期时,如果有其他虚拟机还在服务当前的任务请求,该物理机上的全部虚拟机同时进入高速工作状态[8]。

1.2　系统模型

将移动设备的 CPU 视为本地服务台,建立单一服务台连续工作排队。云端每个虚拟机视为远程服务台,建立多服务台同步多重工作休假排队。结合单一服务台连续工作排队与同步多重工作休假排队,建立系统模型。

假设移动设备任务请求的生成间隔是参数为 $\lambda(0 < \lambda < +\infty)$ 的指数分布。经过本地负载均衡器调度后以概率 p 分配到本地,以概率 $\bar{p} = (1 - p)$ 卸载至云端。

显然,本地服务台任务请求的到达服从参数为 $\lambda_0 = p\lambda(0 < \lambda_0 < \lambda)$ 的 Poisson 过程。假设本地服务台对一个任务请求的服务时间服从参数为 $\mu_0(0 < \mu_0 < +\infty)$ 的指数分布。假设本地只有一个服务台,等待缓存区容量无穷大,假设任务请求的到达过程与服务过程是彼此独立的。因此,本地服务台执行任务请求的过程可看做 M/M/1 排队。

在云端负载均衡器的调度下,分配到云端的任务请求以 $q_i(0 < i \leqslant n)$(n 为云端部署的物理机个数)的概率分配到第 i 个物理机接受服务。对于第 i 个物理机,任务请求的到达服从参数为 $\lambda_i = \bar{p}\lambda q_i$ 的 Poisson 过程。假设第 i 个物理机中部署了 c_i 个相互独立并行工作的虚拟机,当任务请求到达一个物理机时,若有高速空闲状态的虚拟机,则立即接受服务,否则排队等待。一旦物理机服务完成最后一个任务请求,c_i 个虚拟机同时开始随机长度 V_i 的工作休假,其中 V_i 服从参数为 $\theta_i(0 < \theta_i < +\infty)$ 的指数分布。在非工作休假期内,任务请求的服务时间服从参数为 $\mu_{bi}(0 < \mu_{bi} < +\infty)$ 的指数分布,工作休假期服务时间服从参数为 $\mu_{vi}(0 < \mu_{vi} < \mu_{bi})$ 的指数分布。假设所有物理机具有无限缓存,假设任务请求的到达间隔、服务时间和工作休假期的长度相互独立。因此,第 i 个物理机执行任务请求的过程可看做同步多重工作休假 M/M/ c_i 排队[9-10]。

令随机变量 $L_i(t) = l(l = 0,1,\cdots)$ 表示时刻 t 第 i 个物理机中任务请求的数量,称为系统水平。令随机变量 $J_i(t) = j(j = 0,1)$ 表示时刻 t 物理机所处的状态:$j = 0$ 表示物理机处于工作休假状态,$j = 1$ 表示物理机处于正规忙状态。$\{(L_i(t), J_i(t)), t \geqslant 0\}$ 构成一个二维连续时间 Markov 链,其状态空间 $\boldsymbol{\Omega}_i$ 为

$$\boldsymbol{\Omega}_i = \{(0,0)\} \cup \{(l,j): l \geqslant 1, j = 0,1\}$$

令 $\boldsymbol{\pi}_i(l,j)$ 表示稳态下系统水平为 l 且物理机状态为 j 的概率分布。$\boldsymbol{\pi}_i(l,j)$ 定义为

$$\boldsymbol{\pi}_i(l,j) = \lim_{t\to\infty} P\{L_i(t) = l, J_i(t) = j\}, (l,j) \in \boldsymbol{\Omega}_i$$

令 $\boldsymbol{\pi}_i(0) = \boldsymbol{\pi}_i(0,0)$ 表示为稳态下系统水平为 0 的概率向量,令 $\boldsymbol{\pi}_i(l) = (\boldsymbol{\pi}_i(l,0), \boldsymbol{\pi}_i(l,1))$ 表示为稳态下系统水平为 l 的概率向量。二维连续时间 Markov 链 $\{(L_i(t), J_i(t)), t \geqslant 0\}$ 的稳态概率分布 $\boldsymbol{\Pi}_i$ 表示为

$$\boldsymbol{\Pi}_i = (\boldsymbol{\pi}_i(0), \boldsymbol{\pi}_i(1), \cdots)$$

2　系统模型的稳态分析

2.1　稳态条件

根据云计算任务卸载策略建立的系统模型由 1 个M/M/1 排队和 n(n 为云端部署的物理机个数)个同步多重工作休假 M/M/ c_i 排队构成,其中 c_i 为云端部署的第 $i(0 < i \leqslant n)$ 个物理机上虚拟机的个数。为确保整个系统模型达到稳态,需要系统模型中的每一个排队均达到稳态。

本地处理器所对应的 M/M/1 排队的稳态条件

为

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

其中，ρ_0 表示为本地处理器的流通强度。

云端服务器上部署的第 $i(0 < i \leq n)$ 个物理机所对应的同步多重工作休假 M/M/ c_i 排队的稳态条件为

$$\rho_i = \frac{\lambda_i}{c_i\mu_{bi}} < 1 \tag{2}$$

其中，ρ_i 表示云端服务器上部署的第 i 个物理机的流通强度。

结合式(1)与式(2)，给出系统模型处于稳定状态的充分必要条件为

$$\mathrm{Max}\{\rho_0, \rho_1, \cdots, \rho_n\} < 1 \tag{3}$$

2.2 状态转移

根据云端中每个物理机上任务请求的数量 l 与物理机所处状态 j 之间的关系，给出 Markov 链 $\{(L_i(t), J_i(t)), t \geqslant 0\}$ 的状态转移过程，如图 3 所示。

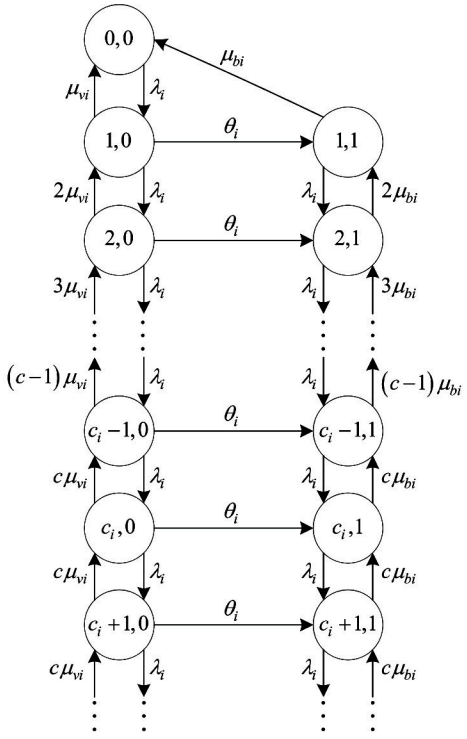


图 3 Markov 过程的状态转移

由图 3 可知，状态转移只发生在相邻的系统水平之间，表明二维 Markov 链 $\{(L_i(t), J_i(t)), t \geqslant 0\}$ 是拟生灭过程。

2.3 稳态分布

令 $\boldsymbol{Q}_i$ 表示 Markov 链 $\{(L_i(t), J_i(t)), t \geqslant 0\}$ 的一步转移率矩阵。根据系统水平划分 $\boldsymbol{Q}_i$ 为若干个子阵。令 $\boldsymbol{Q}_i(x, y)$ 表示系统水平由 $x(x = 0,1,\cdots)$ 到 $y(y = 0,1,\cdots)$ 的转移率子阵。为表述方便，将 $\boldsymbol{Q}_i(x, x-1)$、$\boldsymbol{Q}_i(x, x)$ 和 $\boldsymbol{Q}_i(x, x+1)$ 分别记为 $\boldsymbol{B}_i(x)$、$\boldsymbol{A}_i(x)$ 和 $\boldsymbol{C}_i(x)$。

(1) 当第 i 个物理机中任务请求的数量由 x 变为 $x-1$ 时，转移率子阵为 $\boldsymbol{B}_i(x)$。

当 $x = 1$ 时，若以低速率服务完成任务请求，虚拟机将由低速工作状态变为低速空闲状态，即由状态(1, 0)转换到状态(0, 0)，转移率为 μ_{vi}；若以高速率服务完成任务请求，虚拟机将由高速工作状态变为低速空闲状态，即由状态(1, 1)转换到状态(0, 0)，转移率为 μ_{bi}。$\boldsymbol{B}_i(1)$ 为 2×1 维矩阵，可表示为

$$\boldsymbol{B}_i(1) = \begin{pmatrix} \mu_{vi} \\ \mu_{bi} \end{pmatrix}$$

当 $2 \leq x \leq c_i$ 时，若以低速率服务完成任务请求，虚拟机将由低速工作状态变为低速空闲状态，即由状态 $(x, 0)$ 转换到状态 $(x-1, 0)$，转移率为 $x\mu_{vi}$；若以高速率服务完成任务请求，虚拟机将由高速工作状态变为高速空闲状态，即由状态 $(x, 1)$ 转换到状态 $(x-1, 1)$，转移率为 $x\mu_{bi}$。$\boldsymbol{B}_i(x)$ 为 2×2 维矩阵，可表示为

$$\boldsymbol{B}_i(x) = \begin{pmatrix} x\mu_{vi} & 0 \\ 0 & x\mu_{bi} \end{pmatrix}$$

当 $x > c_i$ 时，若以低速率服务完成任务请求，虚拟机将继续保持低速工作状态，即由状态 $(x, 0)$ 转换到状态 $(x-1, 0)$，转移率为 $c_i\mu_{vi}$；若以高速率服务完成任务请求，虚拟机将继续保持高速工作状态，即由状态 $(x, 1)$ 转换到状态 $(x-1, 1)$，转移率为 $c_i\mu_{bi}$。$\boldsymbol{B}_i(x)$ 为 2×2 维矩阵，可表示为

$$\boldsymbol{B}_i(x) = \begin{pmatrix} c_i\mu_{vi} & 0 \\ 0 & c_i\mu_{bi} \end{pmatrix}$$

(2) 当第 i 个物理机中任务请求的数量 x 保持不变时，转移率子阵为 $\boldsymbol{A}_i(x)$。

当 $x = 0$ 时，第 i 个物理机处于半休眠状态，其上所有虚拟机低速空转。若无任务请求到达，状态

(0, 0)保持不变,转移率为 $-\lambda_i$。$\boldsymbol{A}_i(0)$ 退化为一个数,可表示为

$$\boldsymbol{A}_i(0) = -\lambda_i$$

当 $1 \le x \le c_i$ 时,若 $j = 0$, 第 i 个物理机处于半休眠状态,其上部署的虚拟机部分低速运转部分低速空转。若无任务请求到达也无任务请求离开且休眠定时器未到期,状态 $(x, 0)$ 保持不变,转移率为 $-(\lambda_i + x\mu_{vi} + \theta_i)$; 若休眠定时器到期,物理机由状态 $(x, 0)$ 转换到状态 $(x,1)$, 转移率为 θ_i。若 $j = 1$, 第 i 个物理机处于常规状态,其上部署的虚拟机部分高速运转部分高速空转。若无任务请求到达也无任务请求离开,状态 $(x, 1)$ 保持不变,转移率为 $-(\lambda_i + x\mu_{bi})$。$\boldsymbol{A}_i(x)$ 为 2×2 维矩阵,可表示为

$$\boldsymbol{A}_i(x) = \begin{pmatrix} -(\lambda_i + x\mu_{vi} + \theta_i) & \theta_i \\ 0 & -(\lambda_i + x\mu_{bi}) \end{pmatrix}$$

当 $x > c_i$ 时,若 $j = 0$, 第 i 个物理机处于半休眠状态,其上部署的虚拟机部分低速运转部分低速空转。若无任务请求到达也无任务请求离开且休眠定时器未到期,状态 $(x,0)$ 保持不变,转移率为 $-(\lambda_i + c_i\mu_{vi} + \theta_i)$; 若休眠定时器到期,物理机由状态 $(x, 0)$ 转换到状态 $(x, 1)$, 转移率为 θ_i。若 $j = 1$, 第 i 个物理机处于常规状态,其上部署的虚拟机部分高速运转部分高速空转。若无任务请求到达也无任务请求离开,状态 $(x,1)$ 保持不变,转移率为 $-(\lambda_i + c_i\mu_{bi})$。$\boldsymbol{A}_i(x)$ 为 2×2 维矩阵,可表示为

$$\boldsymbol{A}_i(x) = \begin{pmatrix} -(\lambda_i + c_i\mu_{vi} + \theta_i) & \theta_i \\ 0 & -(\lambda_i + c_i\mu_{bi}) \end{pmatrix}$$

(3) 当第 i 个物理机中任务请求的数量由 x 变为 $x + 1$ 时,转移率子阵为 $\boldsymbol{C}_i(x)$。

当 $x = 0$ 时,第 i 个物理机处于半休眠状态,其上所有虚拟机低速空转。若有任务请求达到,其中一个虚拟机将变为低速工作状态,即由状态(0, 0)转换到状态(1, 0),转移率为 λ_i。$\boldsymbol{C}_i(0)$ 为 1×2 维矩阵,可表示为

$$\boldsymbol{C}_i(0) = (\lambda_i \quad 0)$$

当 $x \ge 1$ 时,若 $j = 0$,第 i 个物理机处于半休眠状态,其上部署的虚拟机既有低速空转又有低速运转。若有任务请求到达,物理机由状态 $(x, 0)$ 转换到状态 $(x+1, 0)$, 转移率为 λ_i。若 $j = 1$, 第 i 个物理机处于常规状态,其上部署的虚拟机部分高速运转部分高速空转。若有任务请求到达,物理机由状态 $(x, 1)$ 转换到状态 $(x + 1,1)$, 转移率为 λ_i。$\boldsymbol{C}_i(x)$ 为 2×2 维矩阵,可表示为

$$\boldsymbol{C}_i(x) = \begin{pmatrix} \lambda_i & \\ & \lambda_i \end{pmatrix}$$

结合以上给出的一步转移率子阵,发现 $\boldsymbol{B}_i(x)(0 < i \le n)$ 和 $\boldsymbol{A}_i(x)(0 < i \le n)$ 从系统水平 c_i 起开始重复,转移率子阵 $\boldsymbol{C}_i(x)(0 < i \le n)$ 从系统水平 1 起开始重复。将重复的 $\boldsymbol{B}_i(x)$、$\boldsymbol{A}_i(x)$ 和 $\boldsymbol{C}_i(x)$ 分别用 $\boldsymbol{B}_i$、$\boldsymbol{A}_i$ 和 $\boldsymbol{C}_i$ 表示,则 Markov 链 $\{(L_i(t), J_i(t)), t \ge 0\}$ 的一步转移率矩阵 $\boldsymbol{Q}_i$ 可表示为分块三对角形式

$$\boldsymbol{Q}_i = \begin{pmatrix} \boldsymbol{A}_i(0) & \boldsymbol{C}_i(0) & & \\ \boldsymbol{B}_i(1) & \boldsymbol{A}_i(1) & \boldsymbol{C}_i & \\ \ddots & \ddots & \ddots & \\ \boldsymbol{B}_i(c_i - 1) & \boldsymbol{A}_i(c_i - 1) & \boldsymbol{C}_i & \\ & \boldsymbol{B}_i & \boldsymbol{A}_i & \boldsymbol{C}_i \\ & & \ddots & \ddots & \ddots \end{pmatrix}$$

拟生灭过程 $\{(L_i(t), J_i(t)), t \ge 0\}$ 正常返的充分必要条件是矩阵二次方程式

$$\boldsymbol{R}_i^2\boldsymbol{B}_i + \boldsymbol{R}_i\boldsymbol{A}_i + \boldsymbol{C}_i = \boldsymbol{0} \tag{4}$$

有最小非负解 $\boldsymbol{R}_i$, 且谱半径 $\mathrm{Sp}(\boldsymbol{R}_i) < 1$。最小非负解 $\boldsymbol{R}_i$ 被称为率阵,可以推出其精确解。

由于转移率子阵 $\boldsymbol{B}_i$、$\boldsymbol{A}_i$ 和 $\boldsymbol{C}_i$ 均为上三角矩阵,因此,率阵 $\boldsymbol{R}_i$ 也为上三角矩阵,表示为

$$\boldsymbol{R}_i = \begin{pmatrix} r_i^{11} & r_i^{12} \\ 0 & r_i^{22} \end{pmatrix}$$

将 $\boldsymbol{R}_i$、$\boldsymbol{B}_i$、$\boldsymbol{A}_i$ 和 $\boldsymbol{C}_i$ 代入矩阵二次方程式(4)中,可得方程组

$$\begin{cases} c_i\mu_{vi}(r_i^{11})^2 - (\lambda_i + c_i\mu_{vi} + \theta_i)r_i^{11} + \lambda_i = 0 \\ c_i\mu_{bi}(r_i^{11}r_i^{12} + r_i^{12}r_i^{22}) + \theta_i r_i^{11} - (\lambda_i + c_i\mu_{bi})r_i^{12} = 0 \\ c_i\mu_{bi}(r_i^{22})^2 - (\lambda_i + c_i\mu_{bi})r_i^{22} + \lambda_i = 0 \end{cases} \tag{5}$$

在 $\rho_i = \dfrac{\lambda_i}{c_i\mu_{bi}} < 1$ 的条件下,求解方程组式(5),得到率阵 $\boldsymbol{R}_i$ 为

$$\boldsymbol{R}_i = \begin{pmatrix} r_i & \dfrac{\theta_i r_i}{c_i\mu_{bi}(1-r_i)} \\ 0 & \rho_i \end{pmatrix}$$

其中，$r_i = \dfrac{1}{2c_i\mu_{vi}}(\lambda_i + c_i\mu_{vi} + \theta_i - \sqrt{(\lambda_i + c_i\mu_{vi} + \theta_i)^2 - 4c_i\lambda_i\mu_{vi}})$

利用所得出的率阵 $\boldsymbol{R}_i$，构建一个方阵 $B[\boldsymbol{R}_i]$ 为

$$B[\boldsymbol{R}_i] = \begin{pmatrix} \boldsymbol{A}_i(0) & \boldsymbol{C}_i(0) & & & \\ \boldsymbol{B}_i(1) & \boldsymbol{A}_i(1) & \boldsymbol{C}_i & & \\ & \ddots & \ddots & \ddots & \\ & \boldsymbol{B}_i(c_i-1) & \boldsymbol{A}_i(c_i-1) & \boldsymbol{C}_i & \\ & & \boldsymbol{B}_i & \boldsymbol{R}_i\boldsymbol{B}_i + \boldsymbol{A}_i \end{pmatrix}$$

由矩阵几何解中的平衡方程和归一化条件可知，拟生灭过程 $\{(L_i(t), J_i(t)), t \geqslant 0\}$ 的平稳分布满足方程组

$$\begin{cases} (\boldsymbol{\pi}_i(0), \boldsymbol{\pi}_i(1), \cdots, \boldsymbol{\pi}_i(c_i))B[\boldsymbol{R}_i] = \boldsymbol{0} \\ (\boldsymbol{\pi}_i(0), \boldsymbol{\pi}_i(1), \cdots, \boldsymbol{\pi}_i(c_i-1))\boldsymbol{e}_1 \\ \qquad + \boldsymbol{\pi}_i(c_i)(\boldsymbol{I} - \boldsymbol{R}_i)^{-1}\boldsymbol{e}_2 = 1 \end{cases} \tag{6}$$

其中，$\boldsymbol{e}_1$ 是 $(2c_i - 1) \times 1$ 的全 1 列向量，$\boldsymbol{e}_2$ 是 2×1 的全 1 列向量，$\boldsymbol{I}$ 是 2×2 的单位阵。

引入增广矩阵 $\boldsymbol{P}_i = \left(B[\boldsymbol{R}_i] \middle| \begin{matrix} \boldsymbol{e}_1 \\ (\boldsymbol{I} - \boldsymbol{R}_i)^{-1}\boldsymbol{e}_2 \end{matrix}\right)$，方程组式(6)可等价表示为

$$(\boldsymbol{\pi}_i(0), \boldsymbol{\pi}_i(1), \cdots, \boldsymbol{\pi}_i(c_i))\boldsymbol{P}_i = (0, 0, \cdots, 0, 1) \tag{7}$$

利用高斯-塞德尔法求解方程式(7)，可以给出 $\boldsymbol{\pi}_i(k)(0 \leqslant k \leqslant c_i)$ 的数值解。

基于矩阵几何解方法，由 $\boldsymbol{\pi}_i(c_i)$ 进一步给出 $\boldsymbol{\pi}_i(k)(k \geqslant c_i + 1)$ 如下：

$$\boldsymbol{\pi}_i(k) = \boldsymbol{\pi}_i(c_i)\boldsymbol{R}_i^{k-c_i}$$

3 系统性能指标

关注云用户响应时间和云系统能耗水平，评估基于半休眠模式的云计算任务卸载策略的性能指标。

任务请求在本地处理器接受服务的平均响应时间 T_{MD} 包括任务请求在本地缓存中的等待时间与在本地处理器上执行的时间。根据 M/M/1 排队模型，可以得到 T_{MD} 的表达式为

$$T_{\mathrm{MD}} = \frac{1}{\mu_0 - \lambda_0} \tag{8}$$

任务请求卸载到云端服务器接受服务的平均响应时间 T_{C} 包括任务请求在云端缓存中的等待时间与在云端服务器上执行的时间。卸载到云端后第 i 个物理机上任务请求的平均响应时间 T_i 的表达式为

$$T_i = \frac{1}{\lambda_i}\sum_{l=c_i+1}^{\infty}(l - c_i)\boldsymbol{\pi}_i(l)\boldsymbol{e} + T_{si}$$

其中，T_{si} 表示任务请求在第 i 个物理机上的平均服务时间，$\boldsymbol{e}$ 为 2×1 的全 1 列向量。

根据卸载到第 i 个物理机上的任务请求到达云端时的状态，分以下 4 种情况讨论 T_{si}。(1) 若第 i 个物理机处于常规状态，当前任务请求接受高速率服务，平均服务时间为 $\dfrac{1}{\mu_{bi}}$；(2) 若第 i 个物理机处于半休眠状态，休眠定时器到期之前缓存中排队等待的所有任务请求及当前任务请求均完成服务，当前任务请求接受低速率服务，平均服务时间为 $\dfrac{1}{\mu_{vi}}$；(3) 若第 i 个物理机处于半休眠状态，休眠定时器到期之前缓存中排队等待的所有任务请求均完成服务，但当前任务请求未服务完成，当前任务请求经历一段低速率服务和一段高速率服务，平均服务时间为 $\dfrac{1}{\mu_{bi}} + \dfrac{1}{\theta_i}$；(4) 若第 i 个物理机处于半休眠状态，休眠定时器到期时，缓存中有排队等待的任务请求未服务完成，当前任务请求接受高速率服务，平均服务时间为 $\dfrac{1}{\mu_{bi}}$。因此，任务请求在第 i 个物理机上的平均服务时间 T_{si} 表示为

$$\begin{aligned} T_{si} = {} & \frac{d_i^1}{\mu_{bi}} + \frac{d_i^0\mu_{vi}}{\mu_{vi} + \theta_i}\left(\frac{\mu_{vi}}{\mu_{vi} + \theta_i}\right)^a \frac{1}{\mu_{vi}} \\ & + \frac{d_i^0\theta_i}{\mu_{vi} + \theta_i}\left(\frac{\mu_{vi}}{\mu_{vi} + \theta_i}\right)^a \left(\frac{1}{\theta_i} + \frac{1}{\mu_{bi}}\right) \\ & + d_i^0\left(1 - \left(\frac{\mu_{vi}}{\mu_{vi} + \theta_i}\right)^a\right)\frac{1}{\mu_{bi}} \end{aligned}$$

其中, $d_i^1 = \sum_{l=1}^{\infty} \pi_i(l,1)$ 表示云端第 i 个物理机处于常规状态的概率, $d_i^0 = \sum_{l=0}^{\infty} \pi_i(l,0)$ 表示云端第 i 个物理机处于半休眠状态的概率, $a = \sum_{l=c_i+1}^{\infty} (l - c_i)\boldsymbol{\pi}_i(l)\boldsymbol{e}$ 表示第 i 个物理机上缓存中任务请求的数量。

令任务请求分配到第 i 个物理机上的概率为 $q_i(0 < q_i < 1)$, 则卸载到云端接受服务的任务请求的平均响应时间 T_C 的表达式为

$$T_C = \sum_{i=1}^{n} q_i T_i \tag{9}$$

云用户的平均响应时间 T 定义为任务请求从进入系统开始直到离开系统所经历的时间长度的平均值。结合式(8)和式(9),给出平均响应时间 T 的表达式为

$$T = pT_{MD} + \bar{p}T_C \tag{10}$$

其中, $p(0 < p < 1)$ 为任务请求在本地处理器接受服务的概率, $\bar{p} = 1 - p$ 为任务请求卸载到云端服务器接受服务的概率。

本地处理器处于工作状态时的平均运行功率表示为 P_{busy}, 处于空闲状态时的平均运行功率表示为 P_{idle}。任务请求在本地处理器接受服务的平均功率消耗 E_{MD} 为

$$E_{MD} = \frac{\lambda_0}{\mu_0}P_{busy} + \left(1 - \frac{\lambda_0}{\mu_0}\right)P_{idle} \tag{11}$$

云端上部署的第 i 个物理机处于半休眠状态时,该物理机维持其上一个虚拟机空转的平均运行功率表示为 P_{vf}^i; 维持其上一个虚拟机运转的平均运行功率表示为 P_{vb}^i。云端上部署的第 i 个物理机处于常规状态时,该物理机维持其上一个虚拟机空转的平均运行功率表示为 P_{bf}^i; 维持其上一个虚拟机运转的平均运行功率表示为 P_{bb}^i[11-12]。任务请求在云端服务器接受服务的平均功率消耗 E_C^i 为

$$\begin{aligned} E_C^i = {} & \sum_{k=0}^{c_i-1}(c_i - k)\pi_i(k,0)P_{vf}^i + \sum_{k=1}^{c_i-1}(c_i - k)\pi_i(k,1)P_{bf}^i \\ & + \left(\sum_{k=0}^{c_i-1} k\pi_i(k,0) + \sum_{k=c_i}^{\infty} c_i\pi_i(k,0)\right)P_{vb}^i \\ & + \left(\sum_{k=1}^{c_i-1} k\pi_i(k,1) + \sum_{k=c_i}^{\infty} c_i\pi_i(k,1)\right)P_{bb}^i \end{aligned} \tag{12}$$

结合式(11)和式(12),给出系统平均功率 E 的表达式为

$$E = pE_{MD} + \bar{p}\sum_{i=1}^{n} q_i E_C^i \tag{13}$$

4　系统实验

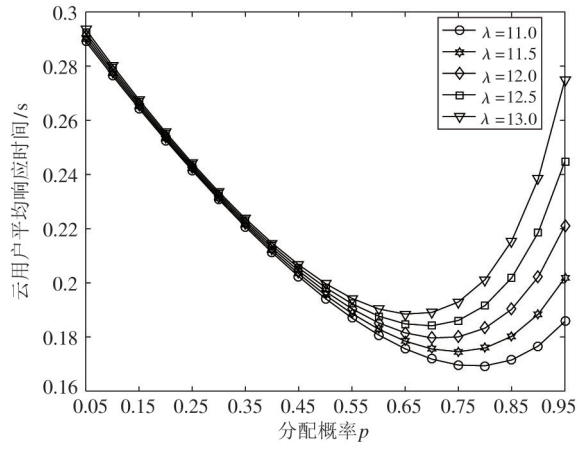
为了进一步进行系统实验研究任务请求初始分配概率与休眠参数对云计算任务卸载策略的影响,进行性能指标变化趋势的系统实验。系统实验的计算机硬件环境为 Intel(R) Core(TM),i7-4790 CPU@3.60 GHz,8.00 GB RAM。软件环境为 Matlab R2016a。

在保证系统模型稳定的条件下的参数设置如表1所示。

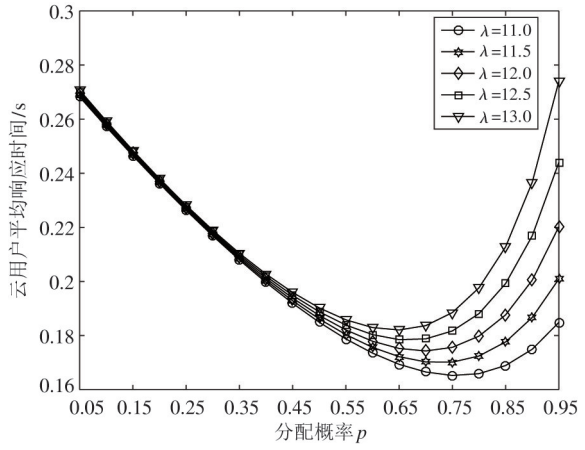
表1　实验参数设置

参数	值
云端物理机数量 n	3 个
不同物理机上部署的虚拟机个数 c_1, c_2, c_3	(2,2,2)个
任务请求分配到不同物理机上的概率为 q_1, q_2, q_3	(0.49,0.22,0.29)
本地处理器的服务率 μ_0	16 个/s
不同物理机的高速服务率 μ_{b1}, μ_{b2}, μ_{b3}	(20, 25, 18)个/s
不同物理机的低速服务率 μ_{v1}, μ_{v2}, μ_{v3}	(7,8,5)个/s
本地处理器处于工作状态时的运行功率 P_{busy}	70 mW
本地处理器处于空闲状态时的运行功率 P_{idle}	20 mW
物理机维持一个虚拟机在半休眠状态下空转的运行功率 P_{vf}^1,P_{vf}^2,P_{vf}^3	(15,17,19) mW
物理机维持一个虚拟机在半休眠状态下低速工作的运行功率 P_{vb}^1,P_{vb}^2,P_{vb}^3	(25,27,29) mW
物理机维持一个虚拟机在常规状态下空转的运行功率 P_{bf}^1,P_{bf}^2,P_{bf}^3	(40,42,44) mW
物理机维持一个虚拟机在常规状态下高速工作的运行功率 P_{bb}^1,P_{bb}^2,P_{bb}^3	(60,62,64) mW

根据表1设定的实验参数,针对不同的休眠参数 θ 和不同的任务请求到达率 λ, 给出云用户平均响应时间 T 随任务请求本地分配概率 p 的变化趋势如图4所示。



(a) θ = 0.5



(b) θ = 1.0

图 4　云用户平均响应时间的变化趋势

对比图 4 (a)和 4 (b)可知,当任务请求本地分配概率 p 和任务请求到达率 λ 一定时,随着休眠参数 θ 的增大,云用户平均响应时间 T 减小。休眠参数 θ 较大时,云端物理机处于半休眠状态的时间长度缩短。这种情况下,云端服务器可以更及时地返回常规状态以高速服务率为任务请求提供服务。因此,云用户平均响应时间 T 减小。

固定任务请求到达率 λ 和休眠参数 θ, 分别观察图 4(a)和图 4(b)可知,随着任务请求本地分配概率 p 的增加,云用户平均响应时间 T 呈现出先下降后上升的趋势。在曲线最低点左侧,任务请求本地分配概率 p 较小,任务请求到达系统后大概率地卸载到云端服务器接受服务,任务请求在云端的响应时间成为云用户平均响应时间 T 的主导因素。随着任务请求本地分配概率 p 的增加,分配到本地的任务请求逐渐增多,堆积在云端的任务请求减少,云用户平均响应时间 T 随之减小。在曲线最低点右侧,任务请求本地分配概率 p 较大,任务请求到达系统后大概率地选择在本地处理器接受服务,任务请求在本地处理器接受服务的响应时间成为云用户平均响应时间 T 的主导因素。随着任务请求本地分配概率 p 的增加,分配到本地的任务请求越来越多,大量的任务请求将会堆积在本地处理器的缓存区,云用户平均响应时间 T 随之增大。

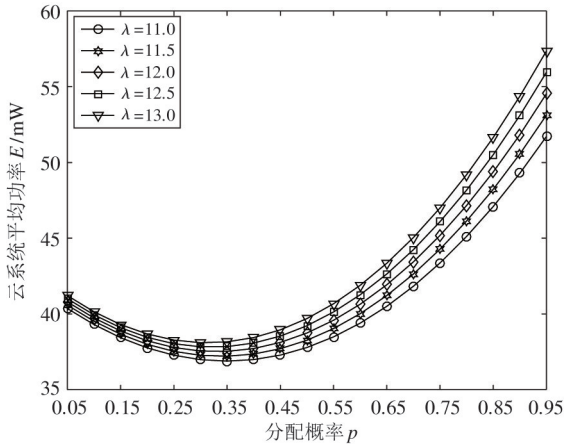
固定休眠参数 θ 和任务请求本地分配概率 p, 分别观察图 4(a)和图 4(b)可知,随着任务请求到达率 λ 的增加,云用户平均响应时间 T 增加。无论是在本地还是云端,随着任务请求到达率 λ 的值增大,系统内到达的任务请求数量也就越多,显然云用户平均响应时间 T 将会增大。

针对不同的休眠参数 θ 和不同的任务请求到达率 λ, 给出云系统平均功率 E 随任务请求本地分配概率 p 的变化趋势如图 5 所示。

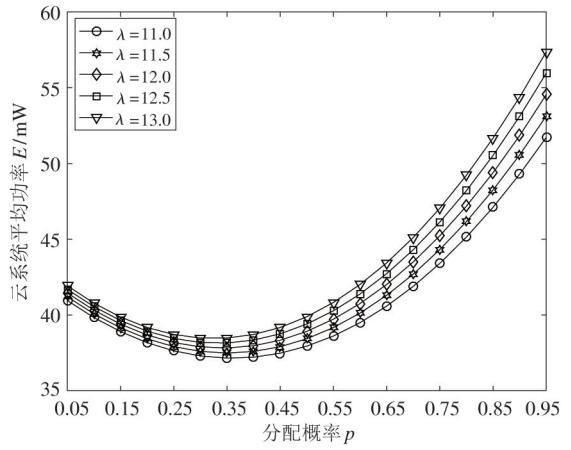
对比图 5(a)和 5(b)可知,当任务请求到达率 λ 一定而任务请求本地分配概率 p 较小时,随着休眠参数 θ 的增大,云系统平均功率 E 略有增大。休眠参数 θ 较大时,云端物理机处于半休眠状态的时间长度缩短,物理机可以更及时地转换到常规状态。处于常规状态的物理机所消耗的功率大于处于半休眠状态的物理机所消耗的功率。因此,云系统平均功率 E 增加。当任务请求到达率 λ 一定而任务请求本地分配概率 p 较大时,卸载到云端的任务请求数量较少,云端物理机有更多的机会持续处于半休眠状态。云系统平均功率 E 与休眠参数 θ 的关系减弱。

固定休眠参数 θ 和任务请求到达率 λ, 分别观察图 5(a)和图 5(b)可知,随着任务请求本地分配概率 p 的增加,云系统平均功率 E 呈现出先下降后上升的趋势。在曲线最低点左侧,任务请求本地分配概率 p 较小,任务请求到达系统后大概率地卸载到云端服务器接受服务。随着任务请求本地分配概率 p 的增加,卸载到云端的任务请求减少,云端物理机有更多的机会进入半休眠状态,云系统平均功率 E 随之减小。在曲线最低点右侧,任务请求本地分

配概率 p 较大,任务请求到达系统后大概率地选择在本地处理器接受服务。随着任务请求本地分配概率 p 的增加,本地处理器提供服务需要的功耗增加,云系统平均功率 E 随之变大。



(a) $\theta = 0.5$



(b) $\theta = 1.0$

图 5　云系统平均功率的变化趋势

固定休眠参数 θ 和任务请求本地分配概率 p,分别观察图 5(a)和图 5(b)还可知,随着任务请求到达率 λ 的增加,云系统平均功率 E 增加。无论是在本地还是云端,随着任务请求到达率 λ 的增大,系统内到达的任务请求数量增多,执行任务请求需要的功耗更多,因此云系统平均功率 E 将会增大。

综合图 4 和图 5 的实验结果可以发现,任务请求本地分配概率是影响系统性能的重要因素。兼顾云用户和云系统的不同要求,在不同的任务请求到达率和相同的休眠参数情况下,优化任务请求本地分配概率。

5　系统优化

为了权衡云用户平均响应时间与云系统平均功率之间的折衷关系,构造系统成本函数 F 为

$$F = \beta_1 T + \beta_2 E \tag{14}$$

其中,β_1 表示云用户平均响应时间 T 对系统成本的影响因子,β_2 表示云系统平均功率 E 对系统成本的影响因子。在实际应用中,对于高响应需求的云系统,可将参数 β_1 的值设置得大些;而对于高节能需求的云系统,则可将参数 β_2 的值设置得大些。

与其他智能优化算法相比,海鸥智能优化算法容易实现,且不需要调整过多的参数[13]。为了加快算法搜索速度,利用 Logistic 映射混沌方法进行海鸥位置初始化,改进海鸥优化算法,优化任务请求本地分配概率。算法中海鸥的位置表示任务请求本地分配概率 p,海鸥的适应值表示系统成本函数 F 的值。算法的主要步骤如下。

步骤 1　初始化海鸥数量 N, 最大迭代次数 $Max_{\text{iterations}}$, 控制海鸥在给定搜索空间中运动行为 A 的频率 f_c, 螺旋攻击形状的相关常数 u、v, 海鸥飞行位置的上边界 Max_{p} 与下边界 Min_{p}。初始化当前迭代次数 $t = 0$。

步骤 2　利用 Logistic 映射方法产生混沌变量初始化海鸥的位置。

$$p_1 = \text{rand}(Max_{\text{p}} - Min_{\text{p}}) + Min_{\text{p}}$$

for　$m = 2:N$

$$p_m = \varepsilon \times p_{m-1} \times (1 - p_{m-1})$$

% $\varepsilon = 4$ 时完全处于混沌状态

endfor

步骤 3　初始化当前最佳海鸥位置 p^* 和适应值 F^*。

$$p^* = p_1$$

$$F^* = p_1(\beta_1 T_{\text{MD}}(p_1) + \beta_2 E_{\text{MD}}(p_1)) + (1 - p_1)(\beta_1 T_{\text{C}}(p_1) + \beta_2 \sum_{i=1}^{n} q_i E_{\text{C}}^i(p_1))$$

步骤 4　根据海鸥迁徙行为和攻击行为更新海鸥位置。

for　$m = 1:N$

$$A = f_c - \left(\frac{t f_c}{Max_{\text{iterations}}}\right)$$

$$D_s(m) = \left| \left(f_c - \left(\frac{t f_c}{Max_{\text{iterations}}}\right)\right) p_m + (2\text{rand} f_c - f_c)(p_1 - p_m) \right|$$

$p_m = D_s(m)(ue^{\omega v})^3 \omega\cos\omega\sin\omega + p_1$

% ω 取(0, 2π)内随机数

endfor

步骤 5 使用系统成本函数 F 计算海鸥的适应值 $F(p_m)$, $m = 1, 2, \cdots, N$。

for $m = 1:N$

$$F(p_m) = p_m(\beta_1 T_{\text{MD}} + \beta_2 E_{\text{MD}}) + (1 - p_m)(\beta_1 T_{\text{C}} + \beta_2 \sum_{i=1}^{n} q_i E_{\text{C}}^i)$$

endfor

步骤 6 找出当前最佳海鸥位置 p^* 和适应值 F^*。

for $m = 1:N$

if $F(p_m) < F^*$

$p^* = p_m$

$F^* = F(p_m)$

endif

endfor

步骤 7 更新迭代次数。

$t = t + 1$ % 更新当前迭代次数

if $t < Max_{\text{iterations}}$

跳转到步骤 4

endif

步骤 8 输出最佳海鸥位置为最优任务请求本地分配概率 p^*；输出最佳适应值为最小系统成本 F^*。

沿用第 4 节系统实验的参数,设定 $\beta_1 = 2$, $\beta_2 = 0.2$, $N = 30$, $Max_{\text{iterations}} = 1000$, $f_c = 2$, $u = 1$, $v = 1$, $Max_{\text{p}} = 0.99999$, $Min_{\text{p}} = 0.00001$。利用改进的海鸥优化算法,给出任务请求本地分配概率的优化结果如表 2 所示。

由表 2 的优化结果可知,对于相同的休眠参数 θ,当任务请求到达率 λ 增大时,受本地处理器处理能力的限制,会有更多的任务请求卸载到云端服务器,因此,最优任务请求本地分配概率 p^* 变小。对于相同的任务请求到达率 λ, 休眠参数 θ 越大,云端服务器处于半休眠状态的时间越短,云端服务器产生的功率消耗越高。为了实现系统成本的最小化,更多的任务请求将在本地处理器接受服务,因此,最优任务请求本地分配概率 p^* 越大越好。

表 2 任务请求本地分配概率的优化结果

任务请求到达率 λ	休眠参数 θ	最优任务请求本地分配概率 p^*	最小系统成本 F^*
11.0	0.5	0.37364	7.8140
11.0	1.0	0.38067	7.8386
11.5	0.5	0.36295	7.8832
11.5	1.0	0.37031	7.9113
12.0	0.5	0.35294	7.9500
12.0	1.0	0.36060	7.9817
12.5	0.5	0.34354	8.0145
12.5	1.0	0.35148	8.0499
13.0	0.5	0.33467	8.0770
13.0	1.0	0.34290	8.1161

6 结 论

综合考虑云用户低响应时间与云系统低功率消耗的要求,根据系统负载调整云端服务器的工作速率,基于半休眠模式,提出了一种云计算任务卸载策略。通过建立本地单一服务台连续工作排队和云端多服务台同步多重工作休假排队,给出了稳态下云用户平均响应时间和云系统平均功率的表达式,并通过实验揭示出不同性能指标之间的折衷关系。从经济学角度出发,构建了系统成本函数,改进传统的海鸥优化算法,给出了任务请求本地分配概率的优化方案。

本文所引入的周期性同步半休眠模式具有良好的节能效果。在未来的研究中,将考虑具有多级适应性的半休眠模式,以进一步提高用户的响应性能。

参考文献

[1] Zhang Y Z, Dong X S, Zhao Y N. Decentralized computation offloading over wierless-powered mobile-edge computing networks[C]// Proceedings of IEEE International Conference on Artificial Intelligence and Information Systems, Fukuoka, Japan, 2020: 137-140

[2] Kuang Z K, Shi Y W, Guo S T, et al. Multi-user offload-

ing game strategy in OFDMA mobile cloud computing system[J]. *IEEE Transactions on Vehicular Technology*, 2019, 68(12): 12190-12201

[3] Malarvizhi N, Priyatharsini S G, Koteeswaran S. Cloud resource scheduling optimal hypervisor for dynamic cloud computing environment[J]. *Wireless Personal Communications*, 2020, 115(1): 27-42

[4] Belgacem A, Beghdad-Bey K, Nacer H, et al. Efficient dynamic resource allocation method for cloud computing environment[J]. *Cluster Computing*, 2020, 23: 2871-2889

[5] 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

[6] Jin S F, Qie X C, Zhao W J, et al. A clustered virtual machine allocation strategy based on a sleep-mode with wake-up threshold in a cloud environment[J]. *Annals of Operations Research*, 2020, 293(1): 193-212

[7] Zhou W C, Fang W W, Li Y Y, et al. Markov approximation for task offloading and computation scaling in mobile edge computing[J]. *Mobile Information Systems*, 2019, 2019: 1-12

[8] 李吉良, 秦兵, 李文江, 等. 融合唤醒阈值与半休眠模式的云虚拟机调度策略[J]. 燕山大学学报, 2020, 44(4): 370-378

[9] 郜燕, 刘文芬, 张建辉. 同步多重工作休假 M/M/*c* 排队系统的性能指标[J]. 工程数学学报, 2012, 29(2): 179-191

[10] 朱翼隽, 徐剑, 周宗好. 多重工作休假的 M/M/*c* 排队系统[J]. 江苏大学学报, 2012, 33(3): 369-372

[11] Schwartz C, Pries R, Tran-Gia P. A queuing analysis of an energy-saving mechanism in data centers[C]// Proceedings of IEEE International Conference on Information Networking, Bali, Indonesia, 2012: 70-75

[12] 孙健, 廖丹, 李可, 等. 基于排队论的异构数据中心性能及能源管理策略[J]. 电子科技大学学报, 2018, 47(2): 161-168

[13] Dhiman G, Kumar V. Seagull optimization algorithm: theory and its applications for large-scale industrial engineering problems[J]. *Knowledge-Based Systems*, 2019, 165: 169-196

Research on semi-sleep mode based task offloading strategy and system performance in cloud computing

Song Jia, Yu Jing, Jin Shunfu

(School of Information Science and Engineering, Yanshan University, Qinhuangdao 066004)

Abstract

Task offloading in cloud computing is a mechanism for migrating intensive mobile applications to cloud environment. By considering the service quality of cloud users and the energy consumption level of cloud system, a periodic synchronous semi-sleep mode is introduced in the cloud and a task offloading strategy is proposed. According to the service rates of virtual machine under different loads, a multi-server queueing model with synchronous multiple working vacation is constructed. Using the quasi-birth-and-death process and matrix-geometric solution method, the average response time of cloud users and the energy consumption of cloud system are derived. System experiments are carried out to evaluate the system performance and verify the validity of the offloading strategy. With the weight factors of different performance measures, a system cost function is constructed. By improving a seagull optimization algorithm, the proposed task offloading strategy in cloud computing is optimized with the minimum system cost.

Key words: cloud computing, task offloading, semi-sleep, working vacation queue, intelligent optimization