文章编号:1000－2367(2018)04－0113－07 **DOI**:10.16366/j.cnki.1000－2367.2018.04.018

QoS 性能约束的云任务调度算法研究

任金霞, 钟小康, 蒋梦倩

(江西理工大学 电气工程与自动化学院,江西 赣州 341000)

摘 要:云计算是目前研究的热点,云计算任务调度中为了在保证用户满意的前提下缩短任务完成时间和提高资源负载均衡性,提出了一种具有 QoS 约束的模拟退火云任务调度算法.首先引入 QoS 约束的贪心策略产生初始解,以最小任务完成时间和最小负载均衡标准差为目标,实行两阶段退火过程,制定两个具有 QoS 约束的新解产生函数,始终处于用户满意的前提下寻找最优分配方案.仿真实验结果表明,该算法能够在保证所有用户都满意的情况下降低任务完成时间并提高资源负载均衡性,是一种顾客和云服务提供商都满意的云任务调度算法.

关键词:云计算;服务质量;贪心策略;资源分配;模拟退火算法

中图分类号:TP393 **文献标志码**:A

随着互联网技术和信息化技术的不断更新,“云”时代已经到来,各种云端产物也如雨后春笋般蓬勃发展.云计算[1－2]就是在这样的大环境下从无到有迅速地壮大起来,具有超大规模、虚拟化、高可靠性、通用性、高可伸缩性、按需服务和廉价的特点,可以说是计算科学的商业实现.云计算利用虚拟化技术抽象物理资源等,用户通过租赁的方式提交申请能够按需获得服务.云服务提供商根据用户提交的任务申请通过调度中心为其分配资源,然而,在云计算明显的动态性特征[3]下,云任务的调度要满足不同任务服务质量(Quality of Service,QoS)需求同时提高云服务系统的吞吐量和资源的利用率是困扰云服务提供商的一个难题.为解决复杂的云任务调度问题,从经典的分布式系统调度算法(贪心算法[4]、Min-Min、Max-Min 等)到启发式智能算法(遗传算法[5－6]、粒子群算法[7－8]、模拟退火算法[9]、蜂群算法[10]、蚁群算法[11]等)纷纷应用到云计算研究中.以上算法大多考虑的是如何缩短任务的完成时间来提高云服务系统的吞吐量或均衡虚拟机的负载来提高资源的利用率,而并没有把所有任务的 QoS 需求(包括完成时间、花费和性能等)考虑在内,更不能充分体现出云计算按需服务的特点.

文中讨论的一种具有 QoS 性能约束机制的云任务调度算法,主要考虑云计算任务的 QoS 性能需求,在满足用户期望的前提下采用改进的两阶段模拟退火云任务调度算法对任务进行调度.仿真实验分析表明,该算法能够确保用户满意,提高云系统的吞吐量和资源的利用率,更好地体现了云计算按需服务的特点.

1 问题描述

云环境下,通过虚拟化技术从大批物理资源中抽象出不同性能的虚拟机组成资源集,用户提交任务申请构成任务集,云平台通过调度策略把任务集中的用户任务分配到资源集中的虚拟机上处理.这个过程对每个用户来说都感觉是自己独自占用一台计算机,从云服务提供商的角度考虑要缩短任务的完成时间(即任务集的最早完成时间)来提高系统的吞吐量,均衡虚拟机的负载来提高资源的利用率,而对于用户来说最为重要的是要满足 QoS 需求.

收稿日期:2017-12-09;**修回日期**:2018-04-26.

基金项目:江西省教育厅科学技术研究项目(GJJ150679)

作者简介:任金霞(1970－),女,山西孝义人,江西理工大学副教授,主要研究方向为智能控制、智能优化算法.

通信作者:钟小康(1988－),男,江西赣州人,主要研究方向为云计算、智能优化,E-mail:zhongxiaokang88@163.com.

1.1 任务模型

用户在云平台上提交任务申请,假设云平台每隔 5 min 对任务集进行作业调度,则每次调度时云平台将用户提交且尚未被分配的任务组成新的任务集进行资源的分配.

$T=\{t_0,t_1,\cdots,t_{n-1}\}$ 表示任务集;其中 n 为云任务的数量,本文假设每个任务都是相互独立的.每个云任务 $t_i=\{t_{\mathrm{id}},t_{\mathrm{length}},t_{\mathrm{data}},t_{\mathrm{rres}},t_{\mathrm{status}}\}$ 表示第 i 个任务的属性,其中 t_{id} 表示任务的 ID,t_{length} 表示任务的总长度,t_{data} 表示任务处理所需的相关数据,t_{rres} 表示任务希望获得的资源属性情况,主要包括资源的计算能力、内存、带宽,由此可量化任务的 QoS 性能需求.t_{status} 表示任务的状态,当用户刚提交任务申请时任务状态为创建(Created),满足条件等待分配资源时为就绪(Ready),资源绑定并等待资源空闲时为等待(Waiting),获得资源处理时为执行(Executed),任务处理完毕时为完成(Finish).

任务的完成时间等于任务的计算时间加上通信时间,设 $T(i,j)$ 表示任务 i 在虚拟机 j 上执行的完成时间,则有:

$$T(i,j)=t^i_{\mathrm{length}}/v^j_{\mathrm{mip}}+t^i_{\mathrm{data}}/v^j_{\mathrm{bw}}. \tag{1}$$

1.2 虚拟机模型

$V=\{v_0,v_1,\cdots,v_{m-1}\}$ 表示云平台为相应任务集提供的虚拟机集合,即虚拟资源集;其中 m 为虚拟机的数量.每台虚拟机可表示为:$v_i=\{v_{\mathrm{id}},v_{\mathrm{mip}},v_{\mathrm{ram}},v_{\mathrm{bw}}\}$,其中 v_{id} 表示虚拟机的序号,v_{mip} 表示虚拟机的计算能力,v_{ram} 表示虚拟机的内存,v_{bw} 表示虚拟机的带宽.每次调度时虚拟机上未执行完的任务不再重新调度,只是记作该虚拟机的初始负载,在同一台虚拟机上执行的任务遵循先进先出原则.本文借用文献[12]所定义的虚拟机综合性能.

定义 1 虚拟机的综合性能 V_{gp} 由虚拟机的各属性及云服务商对各属性的价值认可度计算得出,计算公式如下:

$$V_{gp}=\sqrt{\frac{E_1U_{\mathrm{mip}}+E_2U_{\mathrm{ram}}+E_3U_{\mathrm{bw}}}{E_1+E_2+E_3}}, \tag{2}$$

其中,U_{mip}、U_{ram}、U_{bw} 分别为虚拟机的计算能力、内存和带宽归一化后的值,E_1,E_2,E_3 分别为云服务商对以上 3 种虚拟机属性的价值认可度,且 $E_1+E_2+E_3=1$,用户可根据自己的需要选择合适的.

虚拟机计算能力的归一化公式如下:

$$U_{\mathrm{mip}}=\frac{U_{\mathrm{mip}}-\mathrm{Min}(V_{\mathrm{mip}})}{\mathrm{Max}(V_{\mathrm{mip}})-\mathrm{Min}(V_{\mathrm{mip}})}, \tag{3}$$

其中,V_{mip} 为虚拟机的计算能力,$\mathrm{Max}(V_{\mathrm{mip}})$ 和 $\mathrm{Min}(V_{\mathrm{mip}})$ 分别为虚拟资源集的最大和最小计算能力,其余属性的归一化同理可得.

同样,根据上述公式及任务的期望资源属性的情况 t_{rres} 就可以对任务的 QoS 性能需求进行量化.假定用户提交的某一任务 i 分配到虚拟机 j 上处理,通过(2)式和(3)式计算,若虚拟机 j 的综合性能大于或等于任务 i 的 QoS 性能需求,则认为该分配策略满足用户需求,否则为不满足.

1.3 云任务分配实施

云任务分配分为:用户请求层、虚拟机管理层、用户和数据中心代理层、任务单元执行层.用户请求层对接用户实现与系统的交互,获取用户的任务请求及需求.虚拟机管理层提供虚拟机的创建、分配、销毁和迁移等,对虚拟资源集生命周期进行管理.代理层对用户请求层和虚拟机管理层进行实时的信息交互,动态监测用户的需求、任务的状态及虚拟机的状态,通过调度策略在线协商资源和服务的最优分配.任务单元执行层实现用户任务请求在绑定资源上的最终执行.

2 云任务调度算法

模拟退火算法是 20 世纪 80 年代初期发展起来的一种模仿固体退火过程的智能优化算法,利用 Metropolis 准则并适当地控制温度的下降过程实现模拟退火,理论上能够求出全局最优解.虽然传统的模拟退火算法有原理简单、跳出局部最小、使用灵活等优点,但在复杂多变的云计算环境下对任务调度的运用存在

求解时间长、关键参数选择难、所得解不能兼顾服务质量与时间跨度等缺点.为此,本文根据云计算的特性,先用具有 QoS 约束的任务完成时间贪心策略优化初始解,再把退火计划改进成两个阶段的退火过程,加快求解速度,从而达到改进传统模拟退火算法的目的,使改进后的算法更符合云计算的特性,云任务的调度也更加高效.

2.1　解的组成方式

本文算法所得解的组成方式仿照遗传算法中染色体的编码方式——实数直接编码方式,每个解有多少个基因取决于任务集的任务数量,每个基因的基因值就表示该虚拟资源集中虚拟机的序号.每一个解对应着一种分配方案,例如:假定虚拟机数量为 4 台,任务数为 8 个,则解的基因个数为 8 个,基因值为 4 台虚拟机所对应序号的其中一个;若有一个解为(2,0,3,3,1,2,2,1),则对应的分配方案为:0 号虚拟机执行任务 1,1 号虚拟机执行任务 4 和 7,2 号虚拟机执行任务 0、5 和 6,3 号虚拟机执行任务 2 和 3.

2.2　目标函数

对所得解进行解码可得到一个任务集的分配方案,每台虚拟机上执行的任务各不相同,设第 j 台虚拟机上分配的任务集合 w,则第 j 台虚拟机完成其分配到的任务所用的时间 $F(j)$为:

$$F(j)=\sum_{i=0}^{w}T(i,j), \tag{4}$$

其中,$T(i,j)$ 表示第 j 台虚拟机执行分配在该机上的第 i 个任务所用的时间,由(1) 式求得.

由(4)式可得出所有任务总的执行完成时间 T_F 为:

$$T_F=\underset{j=0}{\overset{m-1}{\mathrm{Max}}}F(j), \tag{5}$$

其中,m 表示虚拟机的数量.

T_F 越小可以提高云系统的吞吐量;则目标函数 1 为:$\min T_F$,用于本文算法的第一阶段退火过程.

同时,对云服务提供商来说为提高资源利用率,虚拟机资源的负载均衡性是非常重要的.本文按文献[8]中所提方法,用虚拟机完成其所分配到的任务所用的时间 $F(j)$来表示虚拟机 j 的负载量,则虚拟资源集的平均负载量 A_L 和负载均衡标准差 B_L 分别为:

$$A_L=\sum_{j=0}^{m-1}F(j)/m, \tag{6}$$

$$B_L=\sqrt{\sum_{j=0}^{m-1}(F(j)-A_L)^2}/m, \tag{7}$$

其中,$F(j)$ 可由(1) 式计算得到;m 表示虚拟机的数量.由此可知,BL 越小,各虚拟机的负载量越接近,资源利用率就越高;因此,目标函数 2 为:$\min B_L$,用于本文算法的第 2 阶段退火过程.

2.3　QoS 约束的贪心策略

在经典的云任务调度算法中贪心算法是一种操作简单、运行速度快的调度策略.它的思想是对任务集的任务依次进行分配资源,希望每次都分配到任务完成时间最短的虚拟机上,实现对时间跨度的贪心.本文把任务 QoS 性能需求融入贪心算法中,在满足任务 QoS 性能需求的前提下对时间跨度实行贪心策略,所得分配方案在任务完成时间 T_F 和负载均衡标准差 B_L 上肯定不是理想的结果,但任务所分配的资源肯定满足任务的 QoS 性能需求.因此,把它作为模拟退火云任务调度算法的初始值对其进行进一步寻优是一个很好的选择.QoS 约束的贪心策略如下.

步骤 1　分别按(1)式和(2)式计算时间跨度矩阵和用户满意度矩阵.

步骤 2　对任务进行资源分配,在用户满意度矩阵中寻找该任务所对应的行值为 1 的元素,如果找到多个符合条件的元素再根据时间跨度矩阵选出当前任务时间跨度最小的那个元素,则该任务分配到序号为选中元素列号的虚拟机,并从任务集中删除该任务.

步骤 3　检查任务集是否为空,如果为否,则转入步骤 2;如果为是,则输出任务集所对应的分配方案,算法结束.

2.4　改进的模拟退火云任务调度算法流程

模拟退火算法由于以 Metropolis 准则来接受新解,使算法能够跳出局部极小值并最终收敛于全局最优

解.在实际的应用中要想得到全局最优解或近似最优解,收敛条件往往要求非常严格,要制定周密的退火计划,每一温度下抽样要足够稳定,合理的新解产生函数、新解接受函数和温度更新函数,选择合适的各项参数等.本文对传统模拟退火算法进行改进,先利用 2.3 节中 QoS 约束的贪心策略获得初始解,再把退火计划制定成两个阶段的退火过程,同时运用两个不同的新解产生函数、新解接受函数和温度更新函数寻找最优解,任务 QoS 性能需求始终贯穿于整个寻优过程,保证所得解对应的分配方案满足所有任务的 QoS 性能需求,确保用户满意.

2.4.1 改进后算法的步骤

改进的模拟退火云任务调度算法如下:

步骤 1 令接受次数 acceptNum=0,分别给定第 1 和第 2 阶段退火过程初始温度 $t(1)$ 和 $t(2)$,sort(cList),和 sort(vList) 对任务集和资源集分别依据任务 QoS 性能需求和资源综合性能按升序排列.

步骤 2 QoS 约束的贪心策略产生初始解 S,令最优解 Best=S,当前解 $S(0)=S$.

步骤 3 进入第 1 阶段退火过程,$S(0)$=Sampling1(Best,$t(1)$)当前解等于抽样 1 在该温度下输出的解,抽样 1(Sampling1)过程见 2.4.2 节.

步骤 4 if(Val(Best)>Val($S(0)$)) then Best=$S(0)$;用(5)式求最优解和当前解的目标函数 1 的值相比较.

步骤 5 $t(1)*=0.999\ 9$ 更新温度.

步骤 6 if($t(1)>t(2)$) then goto 步骤 3;else goto 步骤 7.

步骤 7 令 $p=0$,进入第 2 阶段退火过程,$S(0)$=Sampling2(Best,$t(2)$)当前解等于抽样 2 在该温度下输出的解,抽样 2(Sampling2)过程见 2.4.3 节.

步骤 8 if(Va2(Best)>Va2($S(0)$)) then Best=$S(0)$,$p=0$,else $p++$;用(7)式求最优解和当前解的目标函数 2 的值相比较.

步骤 9 $t(2)*=\exp(-\text{acceptNum}/(\text{vmNum}+\text{cloudletNum}+\text{acceptNum}))$,更新温度.

步骤 10 if($p<$maxstep) then goto 步骤 7;else goto 步骤 11(连续 maxstep 次最优解不变则认为算法收敛).

步骤 11 输出 Best;算法结束.

2.4.2 抽样 1

在第 1 阶段的退火过程中最主要的操作是 Sampling1,它包含了新解产生函数 1 和新解接受函数 1. Sampling1(Best,$t(1)$)抽样 1 过程如下:

步骤 1 令 $q=0$,临时最好解 Best(1)=Best.

步骤 2 利用 Best(1)产生新解 new(1)//新解产生函数 1,借用遗传算法中单点变异规则随机选择 1 个任务,并在资源集中随机选择 1 台综合性能大于或等于任务期望综合性能的虚拟机替换 Best(1)中该任务原分配的虚拟机,得到一个新解 new(1).

步骤 3 计算 Δ=Val(new(1))−Val(Best(1)) if($\Delta<0$||($\Delta>0$&&exp($-\Delta/t(2)$)>random) then Best(1)=new(1),acceptNum++,$q=0$;else $q++$;(新解接受函数 1,以目标函数 1 为标尺接受函数值更小的新解,同时也概率接受更差的新解).

步骤 4 if($q<$maxNum1) then goto 步骤 2,else return Best(1);抽样 1 结束.(当连续 maxNum1 次抽样临时最优解 Best(1)都不变则认为在该温度下抽样稳定)

2.4.3 抽样 2

在第 2 阶段的退火过程中最主要的操作是 Sampling2,它包含了新解产生函数 2 和新解接受函数 2. Sampling2(Best,$t(2)$)抽样 2 过程如下:

步骤 1 令 $q=0$,临时最好解 Best(2)=Best.

步骤 2 利用 Best(2)产生新解 new(2)//新解产生函数 2,借用遗传算法中交叉思想随机选择 2 个任务,若在 Best(2)中这两个任务分配到的虚拟机的综合性能都满足这两个任务的期望值,则交换各自分配到的虚拟机得到一个新解 new(2),否则重新选择.

步骤 3　Δ=Va2(new(2))−Va2(Best(2)) if(Δ<0||(Δ>0&&exp(−100 * Δ/t(2))>random) then Best(1) = new(2) ,q=0,acceptNum++; else q++;(新解接受函数 2,以目标函数 2 为标尺接受函数值更小的新解,同时也概率接受更差的新解).

步骤 4　if(q<maxNum2) then goto 步骤 2,else return Best(2); 抽样 2 结束(当连续 maxNum2 次抽样临时最优解 Best(2)都不变则认为在该温度下抽样稳定).

2.5　算法分析

初始解利用 QoS 约束的贪心策略获得,在满足顾客要求的前提下对任务完成时间进行了初步优化,缩小了第 1 阶段退火过程的搜索范围,加快收敛速度.在第 1 阶段退火过程中,温度以一定的速率减小而新解的产生借用遗传算法变异操作的规则在 QoS 约束下进行全局的搜索,同时以 Metropolis 准则接受新解.当温度降至 t(2),此时全局搜索完成已锁定最优解或近似最优解所在的区间,进入第 2 阶段退火过程进行局部搜索,温度更新与接受新解的次数相关,具有自适应功能,新解的产生则是借助遗传算法交叉操作的思想对当前最好解进行自身单点交换达到周边范围内局部搜索的目的,同时使用目标函数 2 来评价解的好坏是因为在局部搜索时只要保证目标函数 2(负载均衡标准差)最小就能得到理想目标函数 1 值(任务完成时间).纵观全程,不管是初始解的获得,第 1 阶段退火过程还是第 2 阶段退火过程都处在任务 QoS 的约束下,所以本文算法能够在满足顾客要求的前提下减小任务完成时间和负载均衡标准差.

3　仿真实验与分析

本文采用澳大利亚墨尔本大学 Rajkumar Buyya 教授领导团队开发的云仿真器 CloudSim3.0[4] 来进行仿真实验.用 Java 编程语言在 MyEclipse10.7 软件下对 CloudSim 平台的 DatacenterBroker 类进行扩展,写入一个新方法:bindCloudletsToVms ISA(),调用该方法来实现根据本文算法定义的云任务调度策略的模拟实验,通过对本文算法、Min-Min 算法和文献[5]提出的贪心(Greedy)算法相比较来展现本文算法的优越性.

本文设 E_1,E_2,E_3 的值分别为 0.6、0.2、0.2.设置本文算法参数,t(1)=50,t(2)=1,maxStep=200,maxNum1=50,maxNum2=200,贪心算法按文献[5]的方法编写.通过随机发生器随机产生任务集和虚拟资源集,任务长度在[10 000,40 000)区间内,任务所需相关数据在[200,800)区间内,虚拟机计算能力在[100,400) 区间内,内存在[300,1 200)区间内,带宽在[200,800)区间内,任务对资源属性的期望值也由随机发生器按要求随机产生.

3.1　评价指标

云系统对任务进行调度分配的目的是为了增加系统的吞吐量,提高资源的利用率及顾客的满意度.为验证本文算法的性能,模拟实验设置 3 个评价指标,一是由(5)式所得的任务完成时间,二是由(7)式计算所得的负载均衡标准差,三是顾客满意度,假设任务 i 分配到虚拟机 j 上执行,则任务 i 的满意度计算公式如下:

$$S(i)=\begin{cases}1, & v(j)_{gp}\geq t(i)_{gp},\\ 0, & v(j)_{gp}<t(i)_{gp},\end{cases} \tag{8}$$

其中,1 表示满意,0 表示不满意,$v(j)_{gp}$ 和 $t(i)_{gp}$ 分别为虚拟机 j 的资源综合性能和任务 i 的 QoS 性能需求按(2)式和(3)式的量化值,由(1)式计算得出.

顾客满意度 S 的计算公式为:

$$S=\sum_{i=0}^{m-1}S_a(i)/n, \tag{9}$$

其中,n 为任务集中任务的数量.

3.2　实验设计

实验 1　设云系统每隔 5 min 对任务集进行一次调度,虚拟机的数量为 100 台不变,连续调度 5 次,每批任务均为 300 个,重复 10 次实验取平均值,得到如图 1、图 2 和图 3 所示的 3 种算法任务完成时间、虚拟机负载均衡标准差和顾客满意度对比结果.

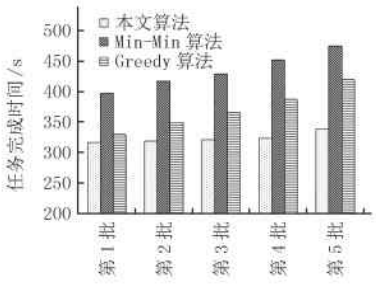

图1 3种算法任务完成时间的比较

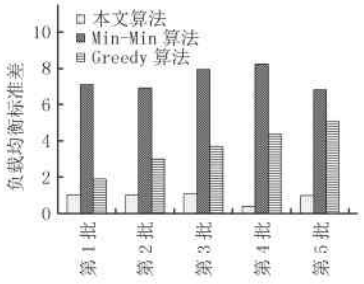

图2 3种算法虚拟机负载均衡标准差的比较

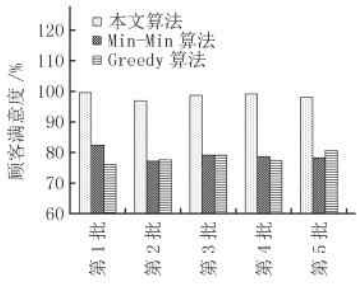

图3 3种算法顾客满意度的比较

实验2 同样假设云系统每隔5 min对任务集进行一次调度,虚拟机的数量为100台不变,连续调度5次,第1批任务为100个,第2批任务为200个,按100个依次增加直到第5批任务为500个,重复10次实验取平均值,得到如图4、图5和图6所示的3种算法任务完成时间、虚拟机负载均衡标准差和顾客满意度对比结果.

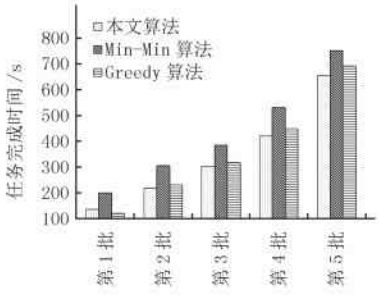

图4 3种算法任务完成时间的比较

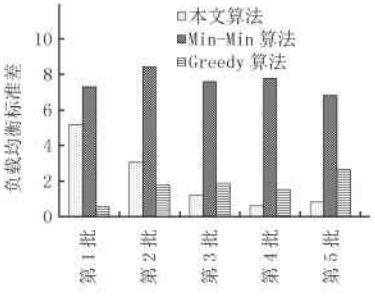

图5 3种算法虚拟机负载均衡标准差的比较

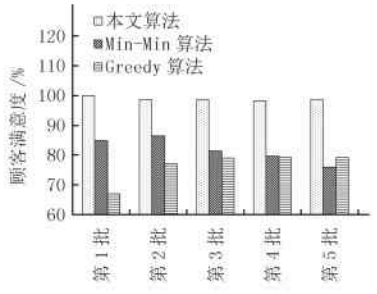

图6 3种算法顾客满意度的比较

3.3 实验结果分析

实验1中虚拟机数量不变,每一批任务集的数量恒定,从图2中可以看出本文算法在负载均衡标准差方面明显优于Min-Min算法,与文献[5]的Greedy算法相比,开始时优势不大,随着调度的深入Greedy算法的负载均衡性越来越差,而本文算法始终处于1以下保持很好的负载均衡性.从另一方面也印证图1的结果:本文算法的任务完成时间比Min-Min算法小近40%,与Greedy算法相比从开始的基本相等到后来优势越来越大.从图3中可以看出本文算法的顾客满意度平均在98%以上比另外两种算法高出近25%.依据本文算法原理,理论上每一次调度顾客满意度都应该为100%,而本文5次调度的结果分别是99.67%、97.00%、98.67%、99.00%、98.00%,其原因是:在模拟实验中由于虚拟机属性数据和任务期望资源属性数据都是随机生成的,5次调度中随机生的5批任务集分别存在1、9、4、3、6个任务的QoS性能需求大于资源集中综合性能最好的虚拟机,导致无法满足任务的需求,所以顾客满意度未能达到100%.而在实际应用中一定存在符合要求的虚拟机,保证100%的顾客满意度.

实验2中虚拟机数量不变,每一批任务集的数量依次增加,由图4和图5可知当每批任务集数量不同时,本文算法的任务完成时间和负载均衡标准差都要比Min-Min算法好;与Greedy算法相比,由于开始调度时任务集数量少而本文算法又处在任务QoS性能需求的约束下分配资源,可以优化的空间小,因此出现图4中前期任务完成时间要略微大和负载均衡性差的情况.随着系统的运行、调度次数的增加、任务数量的增加本文算法的任务完成时间和负载均衡性都要好于Greedy算法,且优越性逐渐加大.图6顾客满意度未达到100%的原因与图3分析的一样.

依据2个实验的结果及以上分析,说明不管任务集变与不变,本文算法能够保证100%的顾客满意度同时在任务完成时间和负载均衡标准差方面也比另外两种算法取得更理想的效果.

4 结　论

本文针对云环境下任务调度问题，以任务 QoS 性能需求为前提，提出了一种具有 QoS 约束的模拟退火云任务调度算法.根据贪心算法的思想引入具有 QoS 约束的贪心策略产生初始解，改进退火计划，制定符合云计算特点处于 QoS 约束下的新解产生函数，选择合理参数进一步优化结果.通过模拟实验证明运用该算法进行云任务调度能够在保证顾客满意的前提下提高系统吞吐量和资源利用率，是一种用户和云服务提供商都兼顾的云任务调度算法.本文的不足之处是未把任务的 QoS 花费需求和能耗需求等考虑在内，下一步的研究工作将是把更多的 QoS 需求加入其中进一步改进算法.

参　考　文　献

[1] Endo P T，Rodrigues M，Goncalves G E，et al.High availability in clouds systematic review and research challenges[J].Journal of Cloud Computing：Advances，Systems and Applications，2016(5)：1-16.

[2] 刘鹏.云计算[M].2 版.北京：电子工业出版社，2011：1-8.

[3] 李功丽，赵晓焱，刘慧.一种云计算数据副本动态管理策略[J].河南师范大学学报(自然科学版)，2015，43(4)：138-143.

[4] 崔雪娇，曾成，徐占然，等.基于贪心算法的云计算资源调度策略[J].微电子学与计算机，2016，33(6)：41-44.

[5] 李腾耀，张凤琴，王梦非.使用遗传算法改进的两阶段云任务调度算法研究[J].小型微型计算机系统，2017，38(6)：1305-1310.

[6] Bahman K，Alireza S，Nima J N.An improved genetic algorithm for task scheduling in the cloud environments using the priority queues：Formal verification，simulation，and statistical testing[J].The Journal of Systems and Software，2017，124：1-21.

[7] 武兴宇，孙磊，胡翠云，等.基于改进粒子群优化算法的虚拟机迁移选择策略研究[J].计算机科学，2015，42(6)：20-23.

[8] 冯小靖，潘郁.云计算环境下的 DPSO 资源负载均衡算法[J].计算机工程与应用，2013，49(6)：105-108.

[9] 张浩荣，陈平华，熊建斌.基于蚁群模拟退火算法的云环境任务调度[J].广东工业大学学报，2014；31(3)：77-82.

[10] 倪志伟，李蓉蓉，方清华，等.基于离散人工蜂群算法的云任务调度优化[J].计算机应用，2016，36(1)：107-112.

[11] 王灿伟.一种云计算环境下的组合寻优调度算法[J].科学技术与工程，2016，16(30)：115-121.

[12] 王波，张晓磊.基于粒子群遗传算法的云计算任务调度研究[J].计算机工程与应用，2015，51(6)：84-88.

Research of cloud task scheduling algorithm with QoS performance constraint

Ren Jinxia，Zhong Xiaokang，Jiang Mengqian

(College of Electrical Engineering and Automation，Jiangxi University of Science and Technology，Ganzhou 341000，China)

Abstract：Cloud computing is a hotspot of current research，the cloud computing task scheduling in order to shorten the task completion time and improve resource load balance on the premise of user satisfacation.A simulated annealing cloud task scheduling algorithm with QoS constraints is proposed in the paper.First greedy strategy of QoS constraints is introduced to produce initial solution.The minimum task completion time and load balance standard deviations as the goal，carry out two-stage annealing process，two generate new solution functions with QoS constraints are formulated，the optimal allocation scheme always is found under the premise of user satisfaction. Simulation experiment results show that the proposed algorithm reduce task completion time and improve resource load balance in the case of ensuring that all users are satisfied. It is a cloud task scheduling algorithm that customers and cloud service providers are satisfied.

Keywords：cloud computing；quality of service；greedy strategy；resource allocation；simulated annealing algorithm

[责任编校　陈留院]