基于云环境的高效任务调度算法

钟志峰 张田田 张 龚 易明星 曾张帆
(湖北大学计算机与信息工程学院 武汉 430062)

摘 要 高效的任务调度是云服务提供商高效处理业务并降低运营成本的关键。针对云环境下的任务调度问题,提出一种贪心模拟退火的新型算法。首先,利用贪心算法求出局部最优解,并用它来初始化所提新型算法的当前最优解及模拟退火算法的初始解;然后,采用模拟退火算法来不断更新当前最优解。实验结果表明,与传统调度算法相比,所提算法能够更快地达到全局收敛,并得到更加稳定的寻优结果,提高了寻优的质量和效率;同时,该算法不仅减少了总任务时间开销,而且使虚拟机的平均资源利用率稳定在99%以上,负载也更加均衡。

关键词 云计算,任务调度,G&SA算法,负载均衡
中图法分类号 TP393 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2018.07.014

Efficient Task Scheduling Algorithm Based on Cloud Environment

ZHONG Zhi-feng ZHANG Tian-tian ZHANG Yan YI Ming-xing ZENG Zhang-fan
(School of Computer and Information Engineering, Hubei University, Wuhan 430062, China)

Abstract Efficient task scheduling is crucial in dealing with business efficiently and cutting down the operating costs for cloud service providers. To improve the performance of task scheduling in cloud environment, this paper proposed a new algorithm, namely greedy simulated annealing (G&SA). Firstly, it finds the local optimal solution by executing the greedy algorithm, which is used to initialize the current optimal solution of the G&SA algorithm and the initial solution of simulated annealing algorithm. Secondly, the current optimal solution is updated by simulated annealing algorithm. As a result, the experiment shows that the G&SA algorithm can achieve global convergence faster compared with the traditional task scheduling algorithm. In addition, the G&SA algorithm not only obtains more stable optimization results and improves the quality and efficiency of optimization, but also reduces the total task time costs. Average resource utilization rate of virtual machine is steady at 99% or more, and the load can be more balanced.

Keywords Cloud computing, Task scheduling, G&SA algorithm, Load balancing

1 引言

云计算是当今计算行业中爆炸性增长的技术之一,它的出现使得将数据和计算迁移到若干系统资源异构的计算资源上成为可能[1]。与管理并利用物理服务器的网格计算不同,云计算基于网络,通过虚拟化技术将虚拟服务器汇聚为逻辑统一的资源池。这种方式降低了用户购买、操作和维护物理计算资源的成本,提高了资源的可用性和灵活性,同时也降低了硬件的复用成本,有助于节约能源[2-3]。

对于云服务提供商来说,在用户数量众多、任务规模庞大、计算资源异构且动态多变的云环境[4]中,高效的任务调度关系到业务处理的综合效率和运营成本[2]。近年来,云计算环境下的任务调度算法逐步成熟,尤其是启发式算法,逐渐被广泛运用。例如,文献[5]采用遗传模拟退火算法对任务调度问题进行研究,算法考虑了不同类型任务的QoS要求,且针对参数维数和数量级的不同做了无量纲处理,有效地完成了基于云环境的资源搜索与分配。针对云环境下的任务调度问题,文献[6]提出了一种自适应人工鱼群算法,该算法采用自适应视野和动态自适应调整步长策略,并改进觅食行为,达到了减小任务执行开销和提高收敛速度的目的。目前,已有的启发式算法各有利弊,如人工鱼群算法(AFSA)的鲁棒性强,具有良好的避免局部极值而取得全局极值的能力,但是寻优结果的精度低,运行速度慢[7];贪心算法(Greedy Algorithm)的求解速度快,局部搜索能力强,但是未从整体上考虑问题,得到的只是局部最优解[8];遗传算法(GA)能够进行多点并行搜索,搜索的广度和随机性强,但是容易过早收敛于局部最优解,并且参数过多,不易于编程实现[9];模拟退火算法(SA)采用Metropolis准则接受新解,具有良好的全局寻优性能,并且鲁棒性强,简单易行[10],但是缺乏记忆功能,对参数比较敏感,需要较多的迭代次数,因此收敛速度慢[11]。

本文以CloudSim为仿真平台,利用该平台将单台服务器虚拟化为多台虚拟机,并构建基于虚拟机的云计算操作平台。将贪心算法局部搜索能力强、求解效率高的优点与模拟退火算法质量高、全局搜索能力强的优点相结合,提出一种贪心模

到稿日期:2017-05-02 返修日期:2017-08-04

钟志峰(1971-),男,博士,副教授,主要研究方向为通信与信息系统及信息系统集成,E-mail:zfzhong@hubu.edu.cn;**张田田**(1991-),女,硕士生,主要研究方向为信号处理与系统集成,E-mail:1165176219@qq.com;**张 龚**(1974-),男,博士,副教授,主要研究方向为智能传感与信息安全,E-mail:2315918@qq.com(通信作者);**易明星**(1994-),男,硕士生,主要研究方向为信号处理与系统集成,E-mail:642283619@qq.com;**曾张帆**(1976-),男,博士,副教授,主要研究方向为信号处理与系统集成。

拟退火任务调度算法。仿真结果表明，与传统的调度算法相比，所提算法的寻优结果更加稳定，并且求解任务调度问题的质量和效率得到了提高；同时，所提算法在任务总时间开销、虚拟机平均资源利用率和系统负载均衡等方面均有所改善。

2　问题描述

本文中的云计算任务调度问题描述如下：将 M 个相互独立的任务分配给 N 台资源异构的虚拟机执行（一般 $N<M$），并使得完成所有任务执行时间开销最少。为了方便研究与仿真，忽略这些资源异构的虚拟机的处理失败率以及虚拟机之间的通信时间等其他因素，只考虑任务长度以及虚拟机资源的性能因素。

设有 N 台资源异构的虚拟资源如下：$VM=\{VM_1,VM_2,\cdots,VM_N\}$，$VM_j(j\in\{1,2,\cdots,N\})$ 表示第 j 个虚拟资源，需要执行 M 个独立任务 $T=\{T_1,T_2,\cdots,T_M\}$，$T_i(i\in\{1,2,\cdots,M\})$ 表示第 i 个任务。用 L_i 表示任务 T_i 的大小，即任务所包含的机器指令条数，单位为 MI（百万条指令数）；用 C_j 表示虚拟机资源 VM_j 的计算性能，单位为 MIPS。

任务 T_i 在虚拟机 VM_j 上的期望执行开销用 ETC（Expected Time to Compute）矩阵[11]表示，该矩阵是一个 $M\times N$ 的矩阵，如下所示：

$$ETC=\begin{bmatrix} E_{11} & E_{12} & \cdots & E_{1j} & \cdots & E_{1N} \\ E_{21} & E_{22} & \cdots & E_{2j} & \cdots & E_{2N} \\ \vdots & \vdots & & \vdots & & \vdots \\ E_{i1} & E_{i2} & \cdots & E_{ij} & \cdots & E_{iN} \\ \vdots & \vdots & & \vdots & & \vdots \\ E_{M1} & E_{M2} & \cdots & E_{Mj} & \cdots & E_{MN} \end{bmatrix} \tag{1}$$

其中，$E_{ij}=L_i/C_j$，$i\in\{1,2,\cdots,M\}$，$j\in\{1,2,\cdots,N\}$，E_{ij} 代表任务 T_i 在虚拟机 VM_j 上执行的时间。

对于一个调度方案 S，将 VM_j 上的负载 Load 定义为其上所有任务的执行时间：

$$Load[j]=\sum_{S(i)=j,\forall T_i\in T} E_{ij} \tag{2}$$

其中，$S(i)=j,\forall T_i\in T$ 表示在调度方案 S 中被分配到虚拟机 VM_j 上的所有任务。利用式(2)直接求解出解决方案 S 的总时间开销 $TotalSpan$ 为：

$$TotalSpan=\max\{Load[j]\mid \forall VM_j\in VM\} \tag{3}$$

单台虚拟机的资源利用率 U 以及每种调度方案中各台虚拟机的平均资源利用率 $AverU$ 可以按照如下定义获得：

$$U(VM_j)=Load(VM_j)/TotalSpan \tag{4}$$

$$AverU=\sum_{i=1}^{N}U(VM_j)/N \tag{5}$$

因此，云计算任务调度的目标就是寻求一种解决方案，使得 $TotalSpan$ 最小，并且虚拟机的平均资源利用率较高。

3　贪心模拟退火算法调度模型

3.1　G&SA 算法

为了实现云环境中任务调度所追求的总任务时间开销较少和系统负载相对均衡的目标，本文采用贪心算法和模拟退火算法组合的优化模型，即利用贪心算法进行任务预调度，得到一个相对理想的调度方案，并将其作为模拟退火算法的初始解以及算法的整体最优解，同时通过模拟退火的方式不断寻求整体最优的调度方案。

贪心算法基于对当前问题的子问题求最优解的原则进行寻优，如式(6)所示。将每个任务分配在最早执行完的虚拟机上；若 T_i 的两种分配方式都能使分配结果最优，则将 T_i 分配在任务量较少的虚拟机上执行，从而实现一种简单的负载均衡。

$$VM_{id}(T_i)=\begin{cases} j-1, & \text{若 } Load[j]+E_{ij}>Load[j-1]+E_{i(j-1)} \\ j-1, & \text{若 } Load[j]+E_{ij}=Load[j-1]+E_{i(j-1)} \\ & \text{且 } TNum(j)>TNum(j-1) \\ j, & \text{否则} \end{cases} \tag{6}$$

其中，$VM_{id}(T_i)$ 表示任务 T_i 被分配到编号为 $VM_{id}(T_i)$ 的虚拟机上，$TNum(j)$ 表示 VM_j 上的任务个数。

$$S_g=[VM_{id}(T_1),VM_{id}(T_2),\cdots,VM_{id}(T_M)] \tag{7}$$

式(7)表示一个解决方案，它由所有任务的分配结果 $VM_{id}(T_i)$ 组合得到，其中 S_g 表示贪心算法的计算结果。由于该结果是一个局部最优解，而模拟退火算法是一种全局优化算法[16]，因此需要采用模拟退火算法对 S_g 进行近一步的处理。

把每种解决方案的 $TotalSpan$ 作为该解决方案的评价函数 Eva，用贪心算法求得的局部最优解 S_g 初始化模拟退火算法的初始解 S_{init}，并根据评价函数公式计算 S_{init} 的评价函数 $Eva(S_{init})$。

$$S_{init}=S_g \tag{8}$$

同时，在模拟退火算法中引入当前最优解 S_{copt}，并用它来记录算法搜寻到的最优解。贪心算法计算结束后，用 S_g 初始化当前最优解 S_{copt}，并记录相应评价函数 $Eva(S_{copt})$。

$$S_{copt}=S_g \tag{9}$$

模拟退火算法在降温过程和迭代过程中利用扰动函数产生一系列的当前解，并比较经历的所有当前解 S_p 和最优解 S_{copt}，根据式(10)更新最优解 S_{copt} 以及对应的评价函数值 $Eva(S_{copt})$。

$$\forall S_p\in S,\ S_{copt}=\begin{cases} S_p, & \text{若 } Eva(S_p)<Eva(S_{copt}) \\ S_{copt}, & \text{其他} \end{cases} \tag{10}$$

3.2　G&SA 降温策略

G&SA 算法的模拟退火过程中，需要通过降温环节来不断寻求最优解。降温策略是温度逐渐降低并达到一个平衡状态的过程，这个过程取决于初始温度 T_0 和温度衰减因子 γ[12]。选择合适的降温策略非常重要，本文的降温策略如下：

$$T_{k+1}=\gamma\times T_k \tag{11}$$

其中，γ 代表温度衰减因子，并且 $0<\gamma<1$。式(11)表示，每一次内部循环完成后，将当前温度乘以衰减因子，以达到降温的目的。

3.3　G&SA 算法扰动

G&SA 算法在模拟退火过程中需要对当前解 S_p 添加扰动，从而产生一个新的状态 S_q，并在每个温度下对当前解进行多次扰动，从而形成一个马尔可夫链，其长度为 ML。算法根据评价函数判断是否用 S_q 替代 S_p，并在每个温度下更新当前最优解 S_{copt}，从而渐渐逼近最佳调度策略。本文采用的扰动模型如式(12)所示：

$$S_q(i)=S_p(i)+(Random()-\alpha)\times Step\times N \tag{12}$$

其中，$i\in\{1,2,\cdots,M\}$，$S_p(i)$ 和 $S_q(i)$ 分别代表当前解和新解中的第 i 个元素；$Random()$ 函数随机产生[0,1]之间的小数；α 为方向因子，用于控制对当前解的扰动方向，一般取[0,1]

之间的小数。本文算法在扰动模型中加入了步长因子 $Step$，其取值范围为(0,1)，用来控制扰动的大小，防止 S_p 跳出最优解所在的范围。

3.4 评价函数及 Metropolis 准则

评价函数又称为目标函数或能量函数，在算法中处于核心地位，用来衡量由式(12)的扰动产生的新解是否为更优。本文算法中的评价函数以总时间开销为准则，计算方式如下：

$$Eva(S_p)=\max\{Load[j]\mid \forall VM_j\in VM\} \tag{13}$$

使用式(12)所示的扰动模型对 S_p 添加随机扰动，得到新状态 S_q，并按照 Metropolis 准则根据对应的评价函数值接受新状态。Metropolis 准则内容如下：

$$A(T_k)\begin{cases}1, & \text{若 } Eva(S_p)\geqslant Eva(S_q)\\ \exp(Eva(S_p)-Eva(S_q)/T_k), & \text{其他}\end{cases} \tag{14}$$

其中，T_k 代表降温过程中降温第 k 次时的温度，$A(T_k)$ 代表当前温度下用 S_q 替换 S_p 的概率。式(14)表示，若满足 $Eva(S_q)\leqslant Eva(S_p)$，则用 S_q 代替 S_p；若 $exp(Eva(S_p)-Eva(S_q)/T_k)>Ran()$，则用 S_q 代替 S_p，否则保留当前解。以某种概率接受次优解，利用这种概率突跳性，在降温策略的控制下逐渐趋近于全局最优解。

3.5 G&SA 算法的收敛性分析

针对本文任务调度算法，根据式(12)给出的扰动模型得出由 S_p 产生 S_q 的概率 $G(T_k)$：

$$G_{pq}(T_k)=\begin{cases}\dfrac{1}{N^M}, & \text{若 } S_q\in D_p\\ 0, & \text{否则}\end{cases} \tag{15}$$

其中，D_p 代表解 S_p 的邻域，$S_q\in D_p$ 表示解 S_q 在解 S_p 的邻域内。

由 Metropolis 准则可知，S_p 接受 S_q 为当前解的概率为 $A_{pq}(T_k)$。由当前解 S_p 转移到 S_q 的一步转移概率 $P_{pq}(T_k)$ 为：

$$P_{pq}(T_k)=\begin{cases}G_{pq}(T_k)A_{pq}(T_k), & p\neq q\\ 1-\sum\limits_{S_l\in D_p,l\neq p}G_{pl}(T_k)A_{pl}(T_k), & p=q\end{cases} \tag{16}$$

将在当前温度 T_k 下进行多次尝试得到解 S_p 的概率定义为 $Q_p(T_k)$。由 G&SA 算法的特性可知，$G(T_k)$ 和 $A(T_k)$ 满足以下条件：

$$\begin{aligned}&\forall T_k>0,\forall S_p,S_q\in S,\\ &\exists m\geqslant 1,\exists l_0,l_1,\cdots,l_m\in S,\\ &\text{使得 } l_0=S_p,l_m=S_q,\\ &\text{且 } G_{l_n l_{n+1}}(T_k)>0,n=0,1,\cdots,m-1;\end{aligned} \tag{17}$$

$$\forall T_k>0,\forall S_p,S_q\in S,G_{pq}(T_k)=G_{qp}(T_k); \tag{18}$$

$$\begin{aligned}&\forall T_k>0,\forall S_p,S_q\in S,\\ &\text{当 } Eva(S_p)>Eva(S_q),A_{pq}(T_k)=1,\\ &\text{否则 } 0<A_{pq}(T_k)<1;\end{aligned} \tag{19}$$

$$\begin{aligned}&\forall T_k>0,\forall S_p,S_q,S_r\in S,\\ &\text{且 } Eva(S_p)\leqslant Eva(S_q)\leqslant Eva(S_r),\\ &A_{pr}(T_k)=A_{pq}(T_k)A_{qr}(T_k);\end{aligned} \tag{20}$$

$$\begin{aligned}&\forall T_k>0,\forall S_p,S_q\in S,\text{且 } Eva(S_p)<Eva(S_q):\\ &\lim_{T\to 0}A_{pq}(T_k)=0;\end{aligned} \tag{21}$$

因此，存在平稳分布 $Q(t)$，其概率分布为：

$$\forall S_p\in S:Q_p(T_k)=\frac{A_{S_{ori}S_p}(T_k)}{\sum\limits_{S_q\in S}A_{S_{ori}S_q}(T_k)} \tag{22}$$

其中，S_{opt} 是全局最优解的集合，$s_{opt}\in S_{opt}$，且：

$$\lim_{T_k\to 0}Q_p(T_k)=\begin{cases}\dfrac{1}{|S_{opt}|}, & S_p\in S_{opt}\\ 0, & \text{其他}\end{cases} \tag{23}$$

式(17)－式(21)给出了 G&SA 算法收敛的充分条件。由式(23)可得：

$$\lim_{T\to 0}(P_T(S_p\in S_{opt}))=\sum_{S_q\in S_{opt}}\frac{1}{|S_{opt}|}=1 \tag{24}$$

因此，只要在当前温度下尽可能多地进行状态转移，使得转移概率达到式(22)的平稳分布，则 G&SA 算法就能以概率 1 收敛于全局最优解 S_{opt}。

4 算法应用及结果分析

为了验证贪心模拟退火(G&SA)算法解决云环境下的资源调度问题的质量与效率，在 MyEclipse2014 环境下以 CloudSim[13] 作为仿真平台来实现 G&SA 算法的仿真研究。G&SA 算法的具体流程如图 1 所示。

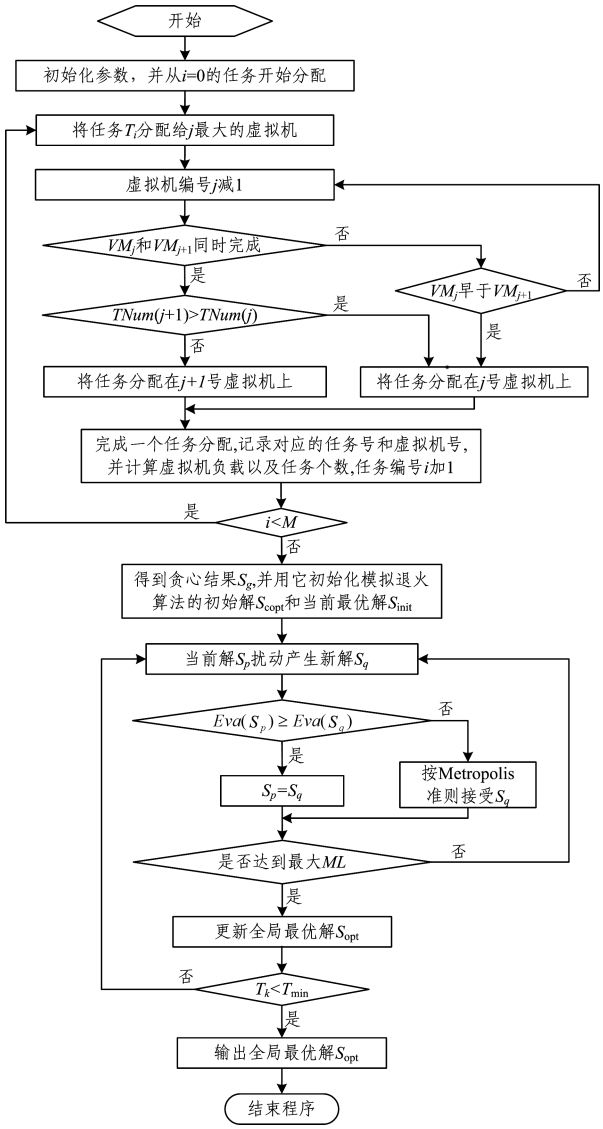


图 1 G&SA 算法的流程图

Fig. 1 Flowchart of G&SA algorithm

本文采用传统的任务调度算法(遗传模拟退火算法(GA&SA)、改进的混沌人工鱼群算法(MCAFSA)、模拟退火算法(SA))和本文算法(G&SA)进行对比实验。虚拟机资源

的参数如表 1 所列。

表 1　虚拟机资源参数

Table 1　Parameters of virtual machine resource

虚拟机号	C/MIPs	内存/GB	带宽/MB
1	885	2	1000
2	803	2	1000
3	802	2	1000
4	815	2	1000
5	957	2	1000

本文算法的主要参数设置如表 2 所列。

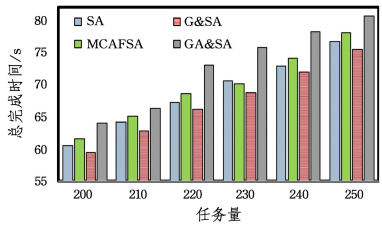
表 2　G&SA 算法参数

Table 2　Parameters of G&SA algorithm

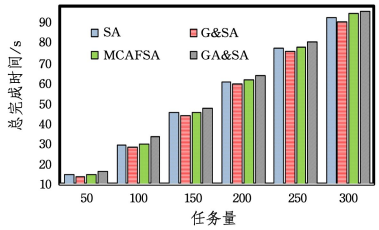
参数	初始值	参数	初始值
ML	[10,100]	T_0	100
T_{min}	0.1	γ	0.9
M	5	α	0.5
N	[50,300]	$Step$	0.23
L	[500,2000]		

在 5 台虚拟机上针对不同任务调度规模进行了仿真，其中任务长度是随机生成的，任务量取值范围为[50,300]，每个任务量重复运行 20 次。仿真中，SA 算法和 G&SA 算法采用相同的参数。算法对比结果如图 2－图 7 所示。

图 2(a)为[50,300]区间的任务总完成时间，图 2(b)为[200,500]区间任务量的总执行时间。图 2(b)中的任务跨度较小，与其他 3 种调度算法相比，相同任务规模下，本文算法在任务总时间开销方面的优势更加明显。在任务量不同时，G&SA 算法能够保持任务的时间开销最小。图 2(b)给出了较大任务跨度的总完成时间，可以从整体上体现出在大、小任务规模调度时，G&SA 算法都能保持任务总耗时最小，一定程度上体现出 G&SA 算法更稳定。总体而言，在大规模任务调度时，本文算法的优势更为明显，表现出了较强的寻优能力。



(a)M 取[50,300] 时的任务总耗时



(b)M 取[200,250]时的任务总耗时

图 2　不同任务量下的任务总耗时

Fig. 2　Total task time with different M

图 3 为 M=200 时 G&SA 算法和 SA 算法的任务总完成时间随着扰动次数变化的曲线图。从图中可以看出，扰动次数较少时，G&SA 算法在 M=200 时的任务完成时间远少于 SA 算法的完成时间，G&SA 算法的优势比较明显，它以贪心算法的计算结果作为最优解的初始解，保证了算法在迭代初期就有很好的收敛效果。而 SA 算法在扰动次数少的情况下收敛效果很不理想，这也体现出 SA 算法受参数影响的缺点；随着扰动次数的增加，其寻优效果逐渐变好。但整体上看，G&SA 算法的总执行开销仍然最小，在改变扰动次数时，G&SA 算法的寻优结果更加稳定，弥补了 SA 算法对参数敏感的缺点。G&SA 算法在迭代次数相对较少时就可以得到优秀的寻优结果，从而加快了算法的收敛速度，可以在一定程度上减少算法耗时。

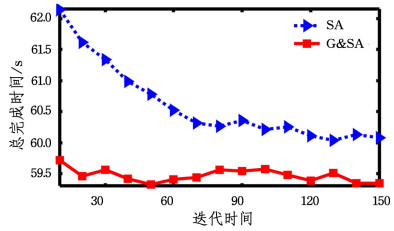


图 3　M=200 时不同扰动次数下任务总耗时

Fig. 3　Total task time with M=200 and different ML

不同任务规模下虚拟机的平均资源利用率如图 4 所示。从图 4 可以看出，3 种对比算法的寻优结果对应的平均资源利用率的波动幅度比较大，而 G&SA 算法的寻优结果对应的虚拟机平均资源利用率相对稳定，并且各个任务规模下的资源利用率都在 99%以上。从整体上来看，G&SA 算法的平均资源利用率明显高于其他 3 种算法，G&SA 算法在负载均衡方面略胜一筹。此外，对应任务量下的时间开销与虚拟机平均资源利用率均是在同一次实验中得出的数据结果。对比图 2 和图 4 可以看出，本文算法在保证减小时间开销的情况下还能提高资源利用率。

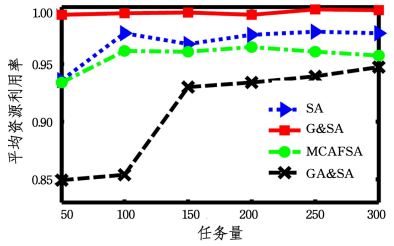


图 4　虚拟机的平均资源利用率

Fig. 4　Average resource utilization of virtual machines

图 5－图 7 分别给出 50,150,300 个任务在 5 台虚拟机上的分布情况。通过比较可以看出，与其他 3 种算法相比，G&SA 算法得出的分配结果在每台虚拟机上的任务数量都比较均匀，从而说明在大、小规模任务调度时，G&SA 算法都能保证资源的负载相对平衡，不会出现由于大量任务集中在某台虚拟机上而导致资源负载失衡的情况。

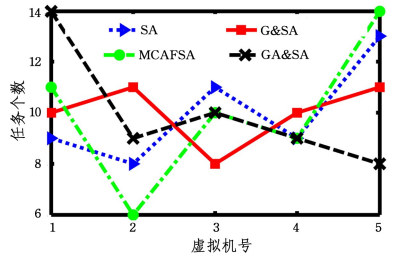


图 5　M=50 时的任务分布

Fig. 5　Task allocation when M=50

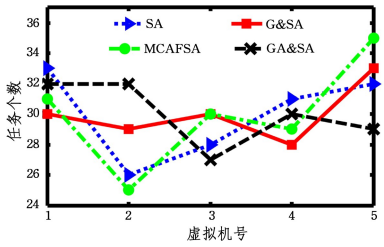


图 6 $M=150$ 时的任务分布

Fig. 6 Task allocation when $M=150$

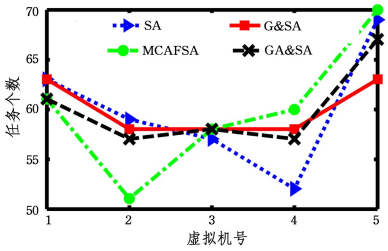


图 7 $M=300$ 时的任务分布

Fig. 7 Task allocation when $M=300$

结束语 针对云环境中虚拟机上的任务调度问题，该文提出了一种贪心模拟退火(G&SA)算法。在 CloudSim 模拟的云平台上进行仿真实验，结果表明，与 SA 算法相比，本文算法在设置不同的退火参数时均能得到更稳定的收敛效果，一定程度上减少了算法的计算时间，大大提高了算法求解任务调度问题的质量和效率。与传统调度算法相比，本文算法在任务总完成时间、虚拟机平均资源利用率、系统负载均衡等方面均有所改善。本文算法只考虑了任务总完成时间、任务长度与虚拟机资源的处理性能，而未考虑其他影响调度算法的因素，如虚拟机之间的通信时间和任务之间的依赖关系等还有待深入研究。

参 考 文 献

[1] YOUNGE A J, HENSCHEL R, BROWN J T, et al. Analysis of Virtualization Technologies for High Performance Computing Environments[C]// IEEE International Conference on Cloud Computing. IEEE Computer Society, 2011: 9-16.

[2] ERGU D, KOU G, PENG Y, et al. The Analytic Hierarchy Process: Task Scheduling and Resource Allocation in Cloud Computing Environment[J]. The Journal of Supercomputing, 2013, 64(3): 1-14.

[3] BOURGUIBA M, El KORBI I, HADDADOU K, et al. Improving Virtual Machines Networking Performance for Cloud Computing[C]// IEEE International Symposium on Integrated Network Management. IEEE, 2013: 513-519.

[4] CHEN H Y. Task Scheduling in Cloud Computing Based on Swarm Intelligence Algorithm[J]. Computer Science, 2014, 41(s1): 83-86. (in Chinese)
陈海燕. 基于多群智能算法的云计算任务调度策略[J]. 计算机科学, 2014, 41(s1): 83-86.

[5] GAN G N, HUANG T L, GAO S. Genetic Simulated Annealing Algorithm for Task Scheduling based on Cloud computing environment[C]// International Conference on Intelligent Computing and Integrated Systems. IEEE, 2010: 60-63.

[6] ZHANG X L. Application of Improved Artificial Fish Swarm Algorithm in Cloud Computing Task Schedule[J]. Electronic Design Engineering, 2017, 25(6): 14-18. (in Chinese)
张晓丽. 改进鱼群算法在云计算任务调度中的应用[J]. 电子设计工程, 2017, 25(6): 14-18.

[7] TIAN L W, TIAN L. A hybrid clustering algorithm based on improved artificial fish swarm[J]. Telkomnika Indonesian Journal of Electrical Engineering, 2014, 12(5).

[8] DONG Z Q, LIU N, ROJAS-CESSA R. Greedy Scheduling of Tasks with Time Constraints for Energy-efficient Cloud-computing Data Centers[J]. Journal of Cloud Computing, 2015, 4(1): 1-14.

[9] XU Y M, LI K L, HU J T, et al. A Genetic Algorithm for Task Scheduling on Heterogeneous Computing Systems using Multiple Priority Queues[J]. Information Sciences, 2014, 270(6): 255-287.

[10] SHI J Y, HU X T, ZOU X B, et al. A Heuristic and Parallel Simulated Annealing Algorithm for Variable Selection in Nearinfrared Spectroscopy Analysis[J]. Journal of Chemometrics, 2016, 30(8): 442-450.

[11] ZHOU L J, WANG C Y. Cloud Computing Resource Scheduling in Mobile Internet Based on Particle Swarm Optimization Algorithm[J]. Computer Science, 2015, 42(6): 279 -281. (in Chinese)
周丽娟, 王春影. 基于粒子群优化算法的云计算资源调度策略研究[J]. 计算机科学, 2015, 42(6): 279-281.

[12] DAMODARANPURUSHOTHAMAN, VELEZ-GALLEGOMARIO C. A Simulated Annealing Algorithm to Minimize makespan of Parallel Batch Processing Machines with Unequal Job Ready Times[J]. Expert Systems with Applications, 2012, 39(1): 1451-1458.

[13] GOYAL T, SINGH A, AGRAWAL A. Cloudsim: Simulator for Cloud Computing Infrastructure and Modeling[J]. Procedia Engineering, 2012, 38(4): 3566-3572.