文章编号:1007－6735(2019)03－0260－07　DOI: 10.13255/j.cnki.jusst.2019.03.008

基于包簇映射的云计算资源分配策略

吕腾飞，　陈世平
（上海理工大学 光电信息与计算机工程学院 上海 200093）

摘要：提出了一种基于包簇映射的云计算资源分配策略。在包、簇概念下，资源可共享，任务调度更为灵活，资源利用率更高。将多目标遗传算法与改进的蚂蚁算法动态融合，提出了一种基于成本最优的云计算资源分配算法。该算法在任务前期利用遗传算法快速随机的全局搜索能力，产生初始信息素，在任务后期通过蚂蚁算法蚂蚁间的信息交流和正反馈机制，寻找资源分配的最优解。实验结果表明，在包、簇概念下，该混合式调度算法能够显著降低云计算系统的任务完成时间和任务执行平均成本，有效减少簇结点的使用数量，提高资源利用率。

关键词：云计算；资源分配；包簇概念；遗传算法；蚂蚁算法；效益模型
中图分类号：TP 393.2　　**文献标志码：**A

Resource Allocation Strategy of Cloud Computing Based on Package-Cluster Mapping

LÜ Tengfei,　CHEN Shiping
(*School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China*)

Abstract: A cloud computing resource allocation strategy based on packet-cluster mapping was proposed. Under the concept of package and cluster, resources can be shared, the task scheduling may become more flexible, and the resource utilization will be raised. Dynamically fusing the multi-objective genetic algorithm with an improved ant algorithm, a cost-optimized cloud computing resource allocation algorithm was provided. In the algorithm, the fast and random global search ability of the genetic algorithm was utilized in the early stage of the task to generate the initial pheromone, and in the later stage of the task, by the ant algorithm, the information among ant colonies was exchanged and a positive feedback mechanism was used to find the optimal solution of the resource allocation. The experimental results show that under the concept of package and cluster, the hybrid scheduling algorithm proposed can significantly reduce the task completion time and task execution cost, effectively reduce the number of cluster nodes used and improve the resource utilization.

收稿日期: 2018−04−09
基金项目: 国家自然科学基金资助项目(61472256)；上海市教委科研创新重点项目(12zz137)；上海市一流学科建设项目(S1201YLXK)
第一作者: 吕腾飞(1993−)，男，硕士研究生．研究方向：云计算、数据仓库．E-mail：1786632701@qq.com
通信作者: 陈世平(1964−)，男，教授．研究方向：计算机网络、云计算、分布式计算．E-mail：chengsp@usst.edu.cn

Keywords: *cloud computing*; *resource allocation*; *package-cluster*; *genetic algorithm*; *ant algorithm*; *benefit model*

云计算是在并行计算、分布式计算、网格计算基础上发展起来的新型计算模型[1-3]。任务调度和资源分配是云计算的两大关键技术，资源分配决定着资源的使用规则，关系到云计算的执行效率和并发处理能力。云计算资源管理的核心是规则和资源分配，即根据确定的资源分配规则，将有限的资源分配给不同的用户，满足用户指定的服务水平目标，优化服务提供架构。

目前，资源分配策略大多是在虚拟机级别上完成的，根据一定的分配策略，实现虚拟机到服务器的映射，即资源管理直接在个体虚拟机与个体服务器间进行[4]。但是，随着数据中心规模的不断扩大，用户需求的动态变化，传统的资源分配方式加大了对虚拟机预留资源的分配，导致所要解决问题规模的扩大，不利于实现虚拟机间空闲资源共享。由于这些资源分配算法的自身局限性，导致解决问题的时间复杂度和空间复杂度比较高，算法整体性能低下，运行效率不理想，数据中心资源利用率不高。

为了突破这些限制，本文引入了一种分层的抽象模型来降解问题的规模，即包、簇概念[5]。通过多目标遗传算法与改进的蚂蚁算法融合[6]，将来自不同用户的需求包映射到合适的资源簇上，来缩短任务时间，提高资源分配效率。在满足需求包的同时，减少资源簇的数目，建立成本模型，评估资源分配过程中数据中心的成本，实现计算绿色、节能[7]。

1　相关工作

云计算中的资源分配问题是多目标优化问题，属于装箱问题，如何高效地实现资源分配是云计算的关键，直接影响到云环境的负载和性能。启发式仿生算法，通过模拟生物进化过程，可以有效解决 NP 完全问题，在众多可行解中，找到问题的最优解，比传统算法更加高效、可靠，有着不可比拟的优越性。

面对用户的弹性要求，云服务提供商如何在满足用户需求的前提下，降低服务成本，健全云计算资源分配机制，减少资源浪费，已经成为国内外云计算学者致力解决的一个研究问题。文献[5]提出了一种基于染色体编码方式和适应度函数的改进遗传算法，仿真结果表明，该算法能更好地适用于大规模任务下的云计算资源调度，但缺点是消耗资源太多和时间太长。文献[8]提出了一种基于蚁群优化的资源分配算法，该算法利用蚁群算法得到一组最优的计算资源，仿真实验证明，分配算法具有更短的响应时间和更好的运行质量，但由于前期信息素匮乏，求解速度较慢。文献[9]提出了以应用性能、保证云应用的服务质量和提高资源利用率为目标的多目标优化模型，并结合最新的神经网络改进粒子群算法对云资源分配求解，但神经网络模型设计的合理性制约着算法的时间复杂度。文献[10]提出了一种云计算环境下基于用户的成本最优存储策略，在满足用户需求的前提下，构建最低服务成本模型，提升服务性能，但该方法采用对文件进行分块、冗余副本机制，将会造成存储空间大量浪费，空间复杂度较差。文献[11]提出了面向云计算任务的资源分配模型，采用层次分析法实现任务资源分配，提高了云计算资源的利用效益，但该方法在构造比较矩阵过程中要维系数以千计的自由变量，可扩展性较差。

在以上研究工作的基础上，本文将多目标遗传算法和改进的蚂蚁算法融合（improved genetic algorithm ant algorithm, IGAAA），该算法在满足不同用户云服务需求的前提下，以节约成本为目标，完成需求包到资源簇的映射，通过建立成本评估模型来评估系统开销，从而减少服务成本，最大化资源利用效率。

2　包簇概念

在传统的虚拟机层面上直接管理资源既不具备扩展性[12]，也不具备资源共享的灵活性。可扩展性问题主要表现在虚拟机中的资源到服务器映射时需要大量的变量维护。灵活性问题主要表现在用户对资源需求的动态要求。

为此，本文引入了新的概念：包和簇，对用户需求和云资源给出多级抽象的递归定义。将用

户的云服务需求抽象成一个需求包，不同用户的需求包汇总在一起，可以进一步抽象为更大的需求包，最上层的所有需求包可以模块化为一个需求池。即定义“包”为虚拟机和其他包的集合。同理，将一个数据中心的计算资源抽象成一个更大的资源池，该资源池由若干计算能力相同或不同的资源簇组成。即定义“簇”为数据中心拓扑位置相近的服务器或更低级别的簇的集合，簇所拥有的资源是其组成部分的资源之和。灵活性可以通过实现需求包到资源簇的映射获得，不同需求包可以映射到相同的资源簇上，实现闲时资源共享。相似地，数据中心资源也可以组成由服务器、簇、簇中簇构成的多级别层次结构。数据中心资源分配问题，将由传统的虚拟机到服务器映射转换成需求包到资源簇映射，当包、簇抽象层级增加时，相关映射问题的规模将进一步下降。

如图1所示，一个跨国公司在上海、伦敦各有一个子公司，其总部设在旧金山。图1中使用了3个包来分别描述总部和子公司的资源需求。其中，总部的需求包有1台防火墙、虚拟机以及3个二级包，分别对应了管理部、财务部和工程部的资源需求。

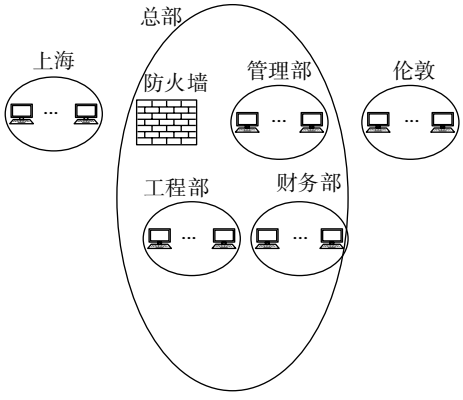

图1 包层次结构

Fig.1 Package structure

图2是在包簇概念下的1个数据中心机架上的服务器情况，图2(a)是1个肥胖树拓扑，当用簇代表每个机架上的服务器后，其拓扑结构简化为图2(b)。将低级别簇聚合成高级别簇后，其拓扑结构又可进一步简化成图2(c)。上层的资源簇是对若干相同或者不同的下层簇的聚合，最下层的资源簇实际上就是映射到机架上的服务器。基于簇的定义，构建资源簇的方式更加灵活。

在包簇概念下，可以通过分层、递归的抽象

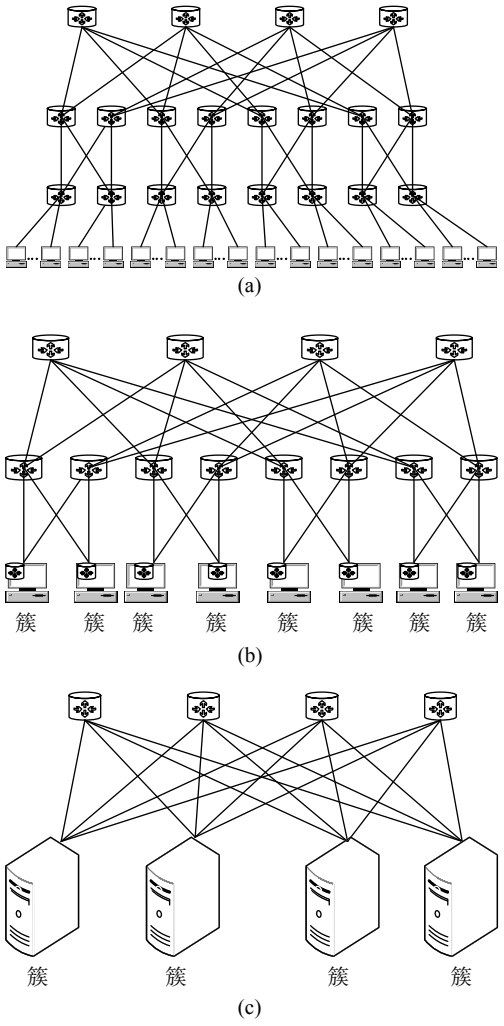

图2 簇层次结构

Fig. 2 Cluster structure

模型，用局部代替整体，降解包到簇映射的问题规模。顶层需求包映射到顶层资源簇上，二级需求包映射到二级资源簇上，更低级别的需求包映射到更低级别的资源簇上。而且用户需求包到资源簇映射的过程是递归重复的，当最底层的虚拟机被映射到机架上的服务器时，停止递归过程，这样已经将用户的所有需求包映射到簇节点上。采用这种“分而治之”的方法，可以将一个复杂度很高的、较难处理的问题简化细分成若干个较低复杂度、可处理的子问题，针对每个子问题，又可以利用本文提出的融合改进算法求解。

在传统的云计算中，利用虚拟化技术可以监控每个计算节点的任务执行情况。在某一时刻如果用户对计算能力的要求超过当前执行节点的最大计算能力，任务将执行失败。但在包、簇概念下，可以很好地解决因为用户需求的弹性变化导致资源分配不能满足用户需求的问题。不再为每

台虚拟机分配一个固定量的资源，而是为这些虚拟机指定一个公共的共享资源池，将这些资源共享的虚拟机以一个需求包的形式整体放在一个服务器集群上。

3　成本评估模型

在云环境中，用户的需求是千差万别的。如一些用户需要很高的 CPU 来保证计算，另一些用户对 CPU 没有特殊要求，但需要更多的存储，用来存储数据文件。CPU 和存储的费用是不同的，不同用户资源的使用时间也是各不相同。为了更好地评估资源分配的成本，本文将从资源使用时间和基础资源成本两个角度构建成本模型。

首先，假设在一个云平台上，某跨国企业最底层有 n 个需求包 $T=\{T_1,T_2,\cdots,T_i,\cdots,T_n\}$，共享数据中心 m 个资源簇 $R=\{R_1,R_2,\cdots,R_j,\cdots,R_m\}$上的资源。影响费用成本的因素有很多，为避免诸如电力、地理位置等因素对成本的影响，在本文中只考虑 CPU、内存、网络带宽对成本的影响。

定义　每个需求包 $T_i=(C_i,M_i,B_i)$，其中，C_i，M_i，B_i 表示该需求对 CPU、内存、网络带宽的具体要求。这些数值来自用户的实际需求，均为准确的数值。这是因为包簇首次映射要求被分配到的资源簇可以满足当前需求包对计算资源的全部要求，首次分配不考虑漂移和共享问题。

对需求包 $T_i=(C_i,M_i,B_i)$到资源簇 R_j 的映射，基于云计算按需分配、按时收费的特点，可以得到 CPU 费用公式为

$$c_c(i,j)=C_b t'_{ij,c}p_{bc}+(C_i-C_b)t''_{ij,c}p_{ec} \quad (1)$$

式中：C_b 表示该资源簇 j 提供的最低 CPU 资源，因为，在包簇概念下，可以有多个包被分配到同一个资源簇上，所以，每个资源簇首先都会为需求包分配一个最低的 CPU 资源；$t'_{ij,c}$是当前需求包对最低 CPU 使用的时间；p_{bc} 是对应最低 CPU 的单位价格；C_i−C_b 表示该需求包超出最低 CPU 资源部分的计算能力；$t''_{ij,c}$是超出资源部分的使用时间；p_{ec} 是超出资源部分的对应单价。

内存资源的收费特点与 CPU 相同，其费用公式为

$$c_m(i,j)=M_b t'_{ij,m}p_{bm}+(M_i-M_b)t''_{ij,m}p_{em} \quad (2)$$

式中：M_b 表示资源簇提供的基础内存大小；$t'_{ij,m}$表示需求包 T_i在资源簇上使用基础内存的时间；p_{bm}表示基础部分的单价；M_i−M_b 表示超出基础内存部分的资源要求；$t''_{ij,m}$是用户对该部分资源的使用时间；p_{em} 是该部分内存资源的单位价格。

数据中心的网络资源是一种特殊资源，对它的精确描述往往还涉及到网络拓扑，有研究表明，数据中心上的大规模云计算可能还会面临带宽瓶颈的问题。数据中心带宽资源的不足将会限制对云中其他资源的最优化利用，并降低包映射到簇时的自由度。本文不考虑资源漂移问题，故对带宽的要求仅满足正常网络传输即可。

本文的目标是在保证所需时间内能完成用户的不同需求，做到最小化资源分配成本，即将每个需求子包分配到特定的资源簇上，使得使用的簇节点数目最少。需求包到资源簇的映射效益模型为

$$R_c=\alpha\sum_{j=1}^{m}\sum_{i=1}^{n}c_c(i,j)+\beta\sum_{j=1}^{m}\sum_{i=1}^{n}c_m(i,j)+\cos\ c_b \quad (3)$$

式中：α 表示当前影响 CPU 费用的权重；β 表示当前影响内存费用的权重；c_b 表示满足正常网络传输带宽的费用；R_c 表示满足 n 个需求包映射到 m 个资源簇的费用。

任务执行时间越少，以及占用更少的数据中心计算资源，则该数据中心资源的利用率越高，因此，本文提出的成本评估模型的目的是最大化包簇映射的效益。

4　云计算资源分配算法设计

4.1　遗传算法与蚂蚁算法的融合

为了解决遗传算法在进化的后期进行大量的无用迭代和蚂蚁算法在进化初期由于缺少信息素进化缓慢的弊端，本文将遗传算法在进化的“适时”阶段融合进蚂蚁算法，即遗传算法与蚂蚁算法的融合改进算法（improved genetic algorithm ant algorithm, IGAAA）。IGAAA 算法的基本思想是吸取遗传算法和蚂蚁算法的优点，克服各自的缺陷，扬长避短，优势互补，融合改进算法总体框架如图 3 所示。

4.2　IGAAA 中遗传算法的设置

遗传算法在解决多目标优化问题时，首先针对问题域中的每种可能情况使用染色体编码方式进行编码，也就是将所有可能存在的资源分配方案进行染色体编码，用来表示需求包到资源簇的映射关系，即每产生一条染色体后代就表示为问题域中的一个可行性解，算法目标首先是得到该问题域所有可行解的候选集，再在可行解候选集

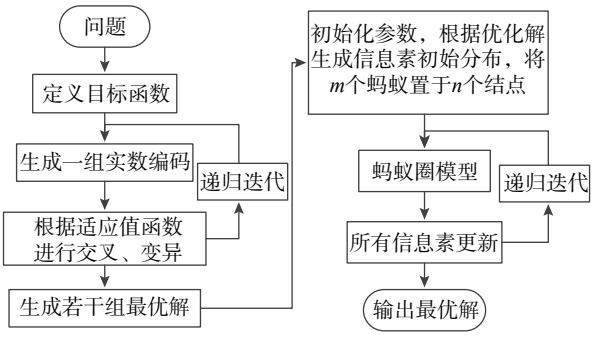

图3 算法框架

Fig. 3 Algorithm framework

中找到最优解，实现需求包与资源簇之间的最优匹配。

步骤1 在IGAAA算法中，采用间接编码方式对染色体编码。初始化阶段，随机地将 n 个需求包映射到 m 个资源簇上。染色体的长度就是需求包的数目。每条染色体上的基因值表示该需求包对应的资源簇。

步骤2 适应度函数用于评估算法优越性、收敛性，本文从成本角度，评估在包、簇概念下的资源分配效益情况。适应度函数

$$f(I)=\frac{1}{R_c} \tag{4}$$

步骤3 根据适应度函数，采用轮回顺序交叉、逆转变异的方法，迭代进化，选择出适应度高的后代。

4.3 IGAAA中蚂蚁算法的改进与衔接

图4为遗传算法与蚂蚁算法进化率随时间的变化趋势图。在NP完全问题中，同等求解规模下，分别采用遗传算法和蚂蚁算法求问题最优解，每经过时间间隔 T，观察进化速率。时间间隔 T 的设定与该问题规模成正比。随着迭代次数的增加，发现在0～3T区间内，遗传算法呈下降趋势，蚂蚁算法呈上升趋势。但是，就整体进化效率而言，遗传算法优于蚂蚁算法。在3T时刻，

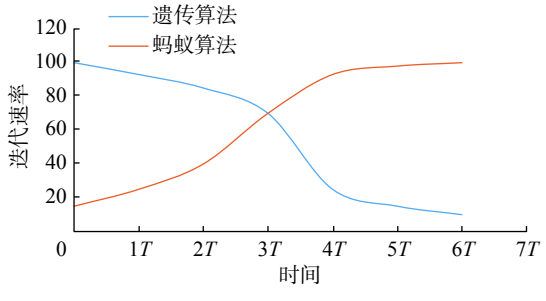

图4 遗传算法与蚂蚁算法进化率–时间关系图

Fig. 4 Evolutionary rate-time relationship between the genetic algorithm and ant algorithm

两算法的进化效率相等；在3T时刻之后，遗传算法出现大量无用的冗余迭代，进化率趋于平缓。由于蚂蚁算法正反馈的特点，信息素浓度的逐渐积累，其进化速率得到显著提高。所以，应在3T时刻之后融入蚂蚁算法。

在本文的问题域中，如何确定3T时刻是解决问题的关键。采用动态融合策略来实现遗传算法与蚂蚁算法的融合。

步骤1 分别设置遗传算法最小迭代次数 $G_{\min}$ 和最大迭代次数 $G_{\max}$。

步骤2 统计遗传算法在迭代过程中子代的进化率，得到整个迭代过程中进化速率的最小值 $G_{\min-r}$。

步骤3 遗传算法在正常求解过程中，第 i 次的迭代记为 G_i，其中，$G_{\min}\leqslant G_i\leqslant G_{\max}$，$G_i\leqslant G_{\min-r}$，说明遗传算法已进入无用迭代时期，求解速度减慢，应在这一时刻引入蚂蚁算法。

蚂蚁圈模型是全局优化较好的蚂蚁算法[13]，结合实际问题，在此采用MMAS(max-min ant system)算法[14]对蚂蚁圈模型改进。初始状态，根据遗传算法得到的信息素，放置蚂蚁，为了获得准确的局部最优解，充分利用蚂蚁算法正反馈寻优特点，将种群中每只蚂蚁经过路径的信息素初始值设为最大值$\tau_{\max}$。在蚂蚁圈中，只有到目标物距离最短的蚂蚁才能进行信息素的修改[15]。为了避免融合算法过早收敛非全局最优解，将各路径的信息素浓度限制在$[\tau_{\min},\tau_{\max}]$之间，超出这个范围的值被强制设为$\tau_{\min}$或$\tau_{\max}$[16]。

通过前期的遗传算法已经得到了一定的路径信息素。所以，在蚂蚁算法初始化阶段，资源簇 R_j 信息素的初始值设置为

$$\tau_j^s(t)=\tau_c+\tau_j^G(t) \tag{5}$$

式中：τ_c就是在蚂蚁圈中设置的$\tau_{\min}$；$\tau_j^G(t)$为遗传算法求解结果转换的信息素。

假如路径(i,j)在 t 时刻信息素轨迹强度为τ_i，蚂蚁 k 在路径(i,j)上留下的单位长度轨迹信息素数量为$\Delta\tau_i^k$，ρ 为轨迹的持久性，$0\leqslant\rho<1$，则轨迹强度的更新方程为

$$\tau_i(t+1)=\rho\tau_i(t)+\sum\Delta\tau_i^k(t) \tag{6}$$

5 实验分析

为了验证本文提出的包、簇概念的科学性、可行性以及基于包、簇概念下云资源分配算法的

收敛性能，现给出仿真实验结果。仿真软件采用 Matlab，CloudSim，设置任务数为 500 个，设置的资源簇为 120 个。通过构建成本评估模型，观察分析簇节点的个数、任务执行时间和成本。对比算法为多目标虚拟机资源分配算法[17]（multi-object virtual machine resource allocation algorithm, MOA）以及文献[6]提出的基于包簇框架的改进多目标遗传算法（improved multi-objective genetic algorithm based on package-cluster, IMOPC）。

本文中所有的服务器资源已经抽象成资源簇概念，用户的需求已经抽象成需求包，为了验证本文算法在同一需求下可以有效减少资源簇的个数，分别采用本文算法和文献[6]中的 IMOPC 算法将 500 台虚拟机部署到 250 台物理服务器。通过仿真实验得到图 5 的结果。实验表明，本文算法部署的簇个数明显要比 IMOPC 算法部署的簇个数要少，这是因为本文算法后期采用了蚂蚁算法，相较单一的遗传算法能够获得更精确的解集。

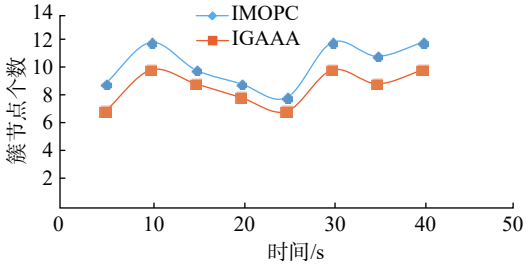

图 5　簇使用个数

Fig.5　Cluster usage

任务完成时间越短，则任务执行过程中对相应资源的占用时间越短，降低了任务执行成本，有效提高了数据中心资源的利用率。为了验证本文算法的性能，将任务数从 100 增加到 500，观察本文算法与对比算法的任务完成时间 y，如图 6 所示。x 为任务数。

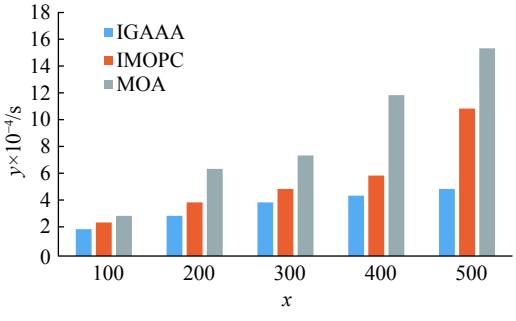

图 6　任务数与任务完成时间关系

Fig.6　Relationship between the task number and task completion time

图 6 中，在相同任务数下，与另外两种算法相比，本文算法的完成时间较少。在任务数小于 300 时，本文算法与 IMOPC 算法任务执行时间相近，但是，随着任务数的逐渐增多，动态融合的 IGAAA 算法的任务完成时间仍然保持在 6×10^4 s 以内，而其他 2 个算法的完成时间已经大幅增长。这主要得益于本文算法前期采用遗传算法获得初始信息素，蚂蚁算法在后期的快速求解能力。通过对比实验可以看出，本文算法能够有效地减少计算资源的投入时间。

云用户在享受便捷云服务的同时，如何降低服务开销是用户普遍关切的问题。对云服务提供商来说，在保证服务质量的同时，降低服务成本也是改进的方向。因此，降低任务执行成本对云用户和云提供商来说都是有益的。在任务数为 200，不同完成时间要求下，分别观察本文算法和对比算法的执行情况，通过建立成本评估模型，评估各算法成本优劣。图 7 给出了仿真实验的结果。

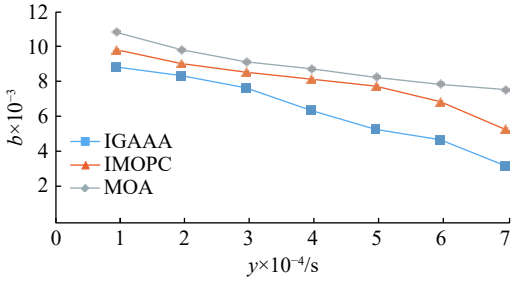

图 7　不同任务完成时间限制下各任务执行成本

Fig.7　Execution cost of each task under the different task completion time limit

由图 7 可知，任务完成时间与费用 b 为近似线性相关。在任一完成时间要求下，本文算法的任务平均执行成本都低于对比算法。当云任务完成时间从 1.0×10^4 增加到 7.0×10^4 时，本文提出的融合改进算法的任务执行成本依旧保持在 3.3×10^4～9.0×10^4，而 IMOPC 算法的任务执行平均成本保持在 5.4×10^4～10.0×10^4，MOA 算法的平均任务执行成本平均在 7.7×10^4～11.0×10^4。分析得出，在同一需求、相同完成时间的条件下，本文提出的融合改进算法的任务执行成本更低，在云环境计算资源分配上具有更好的性能。

在以上 2 个实验的基础上，建立成本评估模型，在任务数为 200，要求任务完成时间为 4×10^4 s 的情况下，取 α=0.7，β=0.3（其中，$\alpha+\beta$=1），将实验参数代入式（4），得到如表 1 所示的实验结果。

表 1 算法性能

Tab.1 Algorithm performance

算法	IGAAA	IMOPC	MOA
效益	1.94×10^{-5}	1.56×10^{-5}	1.46×10^{-5}

在成本评估模型中，采用倒数的形式反映任务效益，即计算结果越大，任务完成时间和任务执行成本越少。由表 1 可以看出，相比其他 2 个对比算法，本文算法具有更高的效益值。

6 结束语

提出了一种基于包簇映射的云计算资源分配策略，通过构建包、簇层次模型，利用分而治之的思想，将庞大、复杂的云资源管理优化问题分解成若干较低复杂度的可以通过递归循环解决的问题。实验证明，利用包、簇概念可有效降解问题规模。为了有效地进行资源分配，将云环境中的计算资源以最低的服务成本通过弹性配置进行按需分配。为了使得计算资源能够合理地按需进行分配，将多目标遗传算法与蚂蚁算法动态融合，在仿真实验中，通过比较任务完成时间、任务执行的平均成本、资源簇的使用个数，衡量改进算法的性能。从实验结果可以看出，在基于包簇框架的资源分配机制中，本文算法在资源分配上取得了较好的效果，相比传统基于虚拟机的分配机制，更大程度地降低了成本。

参考文献：

[1] ARMBRUST M, FOX A, GRIFFITH R, et al. A view of cloud computing[J]. Communications of the ACM, 2010, 53(4): 50–58.

[2] 李乔, 郑啸. 云计算研究现状综述[J]. 计算机科学, 2011, 38(4): 32–37.

[3] OSTERMANN S, IOSUP A, YIGITBASI N, et al. A performance analysis of EC2 cloud computing services for scientific computing[C]//International Conference on Cloud Computing. Berlin: Springer, 2010: 115–131.

[4] 师雪霖, 徐恪. 云虚拟机资源分配的效用最大化模型[J]. 计算机学报, 2016, 36(2): 252–262.

[5] 卢浩洋, 陈世平. 基于包簇映射的云计算资源分配框架[J]. 计算机应用, 2016, 36(10): 2704–2709.

[6] 丁建立, 陈增强, 袁著祉. 遗传算法与蚂蚁算法的融合[J]. 计算机研究与发展, 2003, 40(9): 1351–1356.

[7] 林天高, 马景奕. 基于绿色云计算的能耗优化与资源分配研究[J]. 软件导刊, 2016, 15(5): 7–9.

[8] 聂清彬, 蔡婷, 王宁. 改进的蚁群算法在云计算资源调度中的应用[J]. 计算机工程与设计, 2016, 37(8): 2016–2020.

[9] 赵宏伟, 李圣普. 基于粒子群算法和 RBF 神经网络的云计算资源调度方法研究[J]. 计算机科学, 2016, 43(3): 113–117.

[10] 王宁, 杨扬, 孟坤, 等. 云计算环境下基于用户体验的成本最优存储策略研究[J]. 电子学报, 2014, 42(l): 20–27.

[11] 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): 835–848.

[12] MENG X Q, PAPPAS, ZHANG L. Improving the scalability of data center networks with traffic-aware virtual machine placement[C]//Proceedings of 2010 IEEE INFOCOM. San Diego: IEEE, 2010.

[13] 倪庆剑, 刑汉承, 张志政, 等. 蚁群算法及其应用研究进展[J]. 计算机应用与软件, 2008, 25(8): 12–16.

[14] STÜTZLE T, HOOS H H. MAX-MIN ant system[J]. Future Generation Computer Systems, 2000, 16(9): 889–914.

[15] DORIGO M, GAMBARDELLA L M. Ant colony system: a cooperative learning approach to the traveling salesman problem[J]. IEEE Transactions on Evolutionary Computation, 1997, 1(1): 53–66.

[16] 刘哲, 李风军. 遗传算法与蚂蚁算法的融合在神经网络中的研究[J]. 科技广场, 2012(10): 6–9.

[17] 陈小娇, 陈世平, 方芳. 云计算中虚拟机资源分配算法[J]. 计算机应用研究, 2014, 31(9): 2584–2587.

（编辑：石 瑛）