计算机科学
COMPUTER SCIENCE

混合云下具有交付期约束的众包任务调度算法

严磊，张功萱，王添，寇小勇，王国洪

引用本文

严磊，张功萱，王添，寇小勇，王国洪. 混合云下具有交付期约束的众包任务调度算法[J]. 计算机科学, 2022, 49(5): 244-249.

YAN Lei, ZHANG Gong-xuan, WANG Tian, KOU Xiao-yong, WANG Guo-hong. Scheduling Algorithm for Bag-of-Tasks with Due Date Constraints on Hybrid Clouds[J]. Computer Science, 2022, 49(5): 244-249.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.210300120

混合云下具有交付期约束的众包任务调度算法

严 磊 张功萱 王 添 寇小勇 王国洪
南京理工大学计算机科学与工程学院 南京 210094
(237459461@qq.com)

摘 要 由多个任务组成的众包任务(Bag-of-Tasks,BoT)被广泛应用于各种领域,一般在混合云下执行。与调度问题中传统的截止时间约束不同,交付期约束允许 BoT 应用程序的完成时间超过预先指定的交付期,但会产生延迟惩罚。在这种情况下,为了降低总成本,文中提出了一种高效的和声搜索算法(Efficient Harmony Search,EHS),用于在混合云下优化调度任务。该算法通过随机搜索和微调得到初始任务序列,然后通过改进和声搜索步骤和产生新和声记忆库的方式,在一次搜索过程中可以获得大量优质和声,大大提高了和声搜索的效率,加快了算法的收敛速度。通过不断迭代,获得全局最优解,即使总成本最低的 BoT 应用调度方案。实验结果表明,相比其他算法,该算法在性能上有显著提升,可以有效地降低调度众包任务的总成本。

关键词:混合云;众包任务;交付期;总成本最小化;高效的和声搜索算法
中图法分类号 TP301

Scheduling Algorithm for Bag-of-Tasks with Due Date Constraints on Hybrid Clouds

YAN Lei,ZHANG Gong-xuan,WANG Tian,KOU Xiao-yong and WANG Guo-hong
School of Computer Science and Technology,Nanjing University of Science and Technology,Nanjing 210094,China

Abstract Bag-of-Tasks (BoT) applications consisting of multiple tasks are widely used in various fields. Different from the traditional deadline constraint in scheduling problems,a due date constraint allows the BoT applications to finish more than a predetermined due date,but would result in the tardiness penalty. In this case,in order to reduce the total cost,the efficient harmony search (EHS) algorithm is proposed to optimize the scheduling tasks in hybrid clouds. The algorithm obtains the initial task sequence by random search and fine tuning. By improving the harmony search steps and the way of generating new harmony memory,a large number of high-quality harmony can be obtained in one search process,which greatly improves the efficiency of harmony search and speeds up the convergence speed of the algorithm. Through continuous iteration,the global optimal solution is obtained,that is to say,BoT application scheduling scheme has the lowest total cost. Experimental results show that compared with other algorithms,the proposed algorithm has significant improvement in performance,which can effectively reduce the total cost of scheduling BoT applications.

Keywords Hybrid clouds,Bag-of-Tasks,Due date,Total cost minimization,Efficient harmony search algorithm

1 引言

众包任务应用程序由许多独立的任务组成,这些任务可以在没有同步或没有通信的情况下并行处理。众包任务被广泛应用于计算生物学[1-3]、参数扫描[4-5]、大数据分析[6-7]和并行计算[8-9]等领域。例如,在参数扫描程序中,一个程序可以使用不同的输入配置独立运行多次。而云计算作为一种流行的高性能计算范式[10-12],能够以现收现付的方式交付大量的计算资源[13-14]。换句话说,云用户只需要支付他们实际使用的费用。因此,用户更倾向于在云上提交和执行 BoT 应用程序[15]。但是,私有云无法提供足够的资源,因此可以让用户临时使用公有云的计算资源。为了实现这一目标,一些云供应商提供了混合云解决方案(如 VMware cloud Foundation),将私有云与公有云无缝集成。

本文工作是在图 1 所示的混合云场景中进行的。云提供商接收多个客户的订单,每个订单都有多个要执行的 BoT 应用程序。服务水平协议[16](Service Level Agreement,SLA)建立在每个订单上,它规定了 BoT 应用程序的交付期,以及未能遵守交付期的惩罚率。在传统的调度问题中,要求应用程序在用户指定的截止时间前必须完成。而对于交付期约束而言,允许应用程序在指定的交付期之后完成,但需要通过 SLA 中指定的惩罚率来支付延迟罚款。如果私有云无法提供足够的资源来完成所有任务,那么云提供商就必须将一些任务外包给公有云,但这样会产生额外成本,包括传输数据的

到稿日期:2021-03-11 返修日期:2021-07-20
基金项目:国家自然科学基金(61773206)
This work was supported by the National Natural Science Foundation of China(61773206).
通信作者:张功萱(gongxuan@njust.edu.cn)

成本和使用公有云提供的虚拟机实例的成本。在这种情况下，一个具有挑战性的任务是如何在混合云上优化调度任务，以使总成本（包括数据传输成本、使用公有云提供的虚拟机实例的成本和延迟惩罚）最小化。

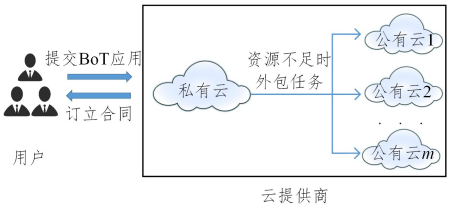

图1　混合云场景

Fig. 1　Hybrid clouds scene

研究混合云环境下 BoT 应用的调度问题有着广泛的应用前景和重要的理论价值。1）研究 BoT 应用程序的调度算法有利于服务器的负载均衡。优化云上的任务调度的一种途径，是把某个负载较重的运算分布到多台节点设备上进行并行处理，等所有的设备都执行完之后，再将结果统一汇总给用户，这样可以增强系统的并行处理能力，同时有利于解决云计算中服务器的负载问题。2）站在云提供商的角度研究混合云环境下 BoT 应用的调度问题具有很大的商业价值。一种设计优良的调度算法不仅能够有效地缩短处理器的空闲时间，提高云计算资源的利用率，还能够降低总成本，增加云提供商的利益。3）在对比和声搜索算法和粒子群优化算法并吸收它们的优点之后，本文提出的高效的和声搜索算法能够超越之前的算法，具有较高的理论价值。

在之前的诸多文献中，已经有大量的学者研究了混合云环境下的 BoT 应用调度问题。文献[17]构造了两种有效的启发式算法来解决调度问题，同时考虑了计算和数据传输的成本。文献[18]与其他将私有云视为资源池的文献不同，它将私有云视为多个离散的物理机器，并设计了一种贪婪的启发式算法来调度这些物理机器之间的任务。文献[19]研究了在不同的运行时间下，调度 BoT 应用程序的问题，提出了一种估计任务运行时间的方法，根据估计的任务持续时间更新调度结果。文献[20]指出合理分配资源可以在一定程度上解决服务器的负载波动问题，同时还可以保证服务质量，其目标是最小化分配虚拟机实例的花费。文献[21]提出了一种高效的粒子群优化算法和一种将粒子映射到解的映射算子，并设计了一种快速任务分配方法来计算解的目标值。文献[22]提出了一类基于有效的任务重调度策略的启发式算法，该策略考虑了从私有云向公有云传输数据的成本，以进一步降低总成本。文献[23]从 SaaS 提供商的角度解决了具有交付期约束的 BoT 调度问题。SaaS 提供商从公有云租用固定数量的虚拟机，并向客户提供服务，其关键问题是如何将由 BoT 应用程序组成的请求以最小的总成本分配给虚拟机。每个 BoT 应用程序都有一个软的截止日期（即交付期）和一个固定的截止日期。换言之，每个 BoT 应用程序可以在其软截止日期后完成，但不允许超过其固定的截止日期。基于著名的 max-min 策略，Stavrinides 等[23]提出了一种有效的启发式算法来解决这个调度问题。然而，这种方法假设在公有云上调度 BoT 应用程序，并且具有固定数量的虚拟机，因此无法处理本文中考虑的混合云场景。文献[24]提出了一种任务范围调节算法和基于预测分析的在线任务分配算法，用于解决时空众包在线任务分配问题，并且两种算法相结合可以有效地提升任务分配的总效用。文献[25]提出了一种改进的软件众包模块分配算法，用于加快众包任务分配的效率，与其他算法相比，该算法可显著提升算法的适配值。文献[26]提出了一种基于预测算法的在线空间众包任务分配策略，可有效提升任务分配的效率与质量。

针对混合云环境下具有交付期约束的 BoT 应用的调度问题，目前的研究工作尚未形成一个完整有效的解决方案。本文的创新性在于，提出了一种混合云环境下具有交付期约束的 BoT 应用调度算法，即高效的和声搜索算法。该算法继承了和声搜索算法和遗传算法的优点，优化了和声搜索步骤，提高了算法的有效性和稳定性。在同样的混合云环境下，该算法可以得到更优的调度方案，有效地降低了总成本。

2　问题描述

下文将详细介绍考虑问题的优化模型，其中应用场景可以广泛应用于其他现有方法。首先建立一个混合云环境，包括一个私有云CP_0和 m 个公有云$CP_1,CP_2,\cdots,CP_m$。私有云可以提供 k 个虚拟机类型，分别为$VM_1,\cdots,VM_k$。每个虚拟机$VM_q(q=1,2,\cdots,k)$有两个属性CPU_q和Mem_q，分别表示该虚拟机的 CPU 数量和内存容量。当私有云无法完成任务时，任务必须外包给公有云，并创建应用程序所需的虚拟机类型的实例。通过混合云的构建，可以等价地认为公有云也提供了 k 个虚拟机类型，分别为$VM_1,\cdots,VM_k$。用户可以免费地使用私有云上的资源，但是使用公有云上的资源是需要收费的。每个公有云$CP_h(h=1,2,\cdots,m)$都有各自的带宽b_h（以每秒千兆字节为单位）、每千兆字节的入站价格 pi_h 和每个$VM_q\in CP_h$每单位时间的价格P_{hq}。

假设云提供商有 n 个 BoT 应用程序需要执行，分别是$a_1,a_2,\cdots,a_n$，每个应用程序$a_i(i=1,2,\cdots,n)$包含D_i的数据量，并由T_i个任务$t_{i1},t_{i2},\cdots,t_{iTi}$组成。在这个交付期约束的调度问题中，每个应用程序$a_i$都有一个完成任务的交付期$d_i$和一个惩罚率$pb_i$。假设任务$t_{ij}(i=1,2,\cdots,n;j=1,2,\cdots,T_i)$（即应用程序$a_i$上的任务$t_j$），如果$t_{ij}$计划在 VM 实例VM_q上执行，那么$r_{ijq}(i=1,2,\cdots,n;j=1,2,\cdots,T_i;q=1,2,\cdots,k)$就表示其持续时间。任务的执行时间通常是以小时为单位计量的，因此虚拟机的准备时间相比任务的执行时间可以忽略不计。在制定调度框架时，将时间单位设为 1 s，将整个时间段划分为若干个粒度为 1 s 的时隙。本文规定，同一时间只在一个虚拟机上执行一个任务，且按照安排好的任务序列顺序执行，不允许抢占。私有云CP_0的可用资源有限，使用CPU^*和Mem^*分别表示其 CPU 和内存容量。因此对于任何时隙 s，私有云消耗的 CPU 总量和内存总量不能超过CPU^*和Mem^*。我们定义时隙的最大值$S=\max\{d_1,d_2,\cdots,d_n\}$。本文认为，公有云拥有无限的资源。

我们将惩罚建模为惩罚率的线性函数，如果应用程序的最大完工时间为c_i，则惩罚pt_i可以确定为：

$$pt_i=\begin{cases}0, & \text{if } c_i\leqslant d_i\\ (c_i-d_i)*pb_i, & \text{if } c_i>d_i\end{cases} \tag{1}$$

其中，c_i-d_i是拖延时间。我们用$S=\max\limits_{i=1,2,\cdots,n}\{c_i\}$表示所有任务的最大时隙数，其中$c_i$是其所有任务完成时间中的最大值：

$$c_i=\max_{j=1,2,\cdots,T_i}\{c_{ij}\} \tag{2}$$

其中，c_{ij}表示任务t_{ij}的完成时间。如果将t_{ij}分配给CP_0并映射到VM_q，其开始时间为st_{ij}，则c_{ij}如式(3)所示：

$$c_{ij}=st_{ij}+r_{ijq} \tag{3}$$

对于分配给CP_h并映射到VM_q上的任务t_{ij}，其完成时间c_{ij}如式(4)所示：

$$c_{ij}=\frac{D_i}{b_h}+r_{ijq} \tag{4}$$

其中，$\frac{D_i}{b_h}$表示数据的传输时间。

我们定义了3个二元决策变量x_{ijq}，y_{ijh}和z_{ijs}，用于问题的求解。如果任务t_{ij}映射到VM_q，则$x_{ijq}=1$，否则$x_{ijq}=0$。同样，如果任务t_{ij}外包给CP_h，则$y_{ijh}=1$，否则$y_{ijh}=0$。$z_{ijs}=1$表示任务t_{ij}在CP_0的时隙s开始执行，否则$z_{ijs}=0$。因此，c_i可以用决策变量表示：

$$c_i=\max_{i,q,k}\left\{(x_{ijq}z_{ijs}(s+r_{ijq}))\cup\left(\left(x_{ijq}y_{ijh}\left(\frac{D_i}{b_h}+r_{ijq}\right)\right)\right)\right\} \tag{5}$$

本文在y_{ijh}的基础上定义了一个辅助变量v_{ih}。$v_{ih}=1$表示至少有一个属于a_i的任务被分配给CP_h(即，$\exists j\in\{1,2,\cdots,T_i\}:y_{ijh}=1$)，否则$v_{ih}=0$。

将考虑的调度问题表示为式(6)，且本文的目的是使总成本(Total Cost，TC)最小化：

$$TC=\sum_{i=1}^{n}pt_i+\sum_{h=1}^{m}\sum_{i=1}^{n}v_{ih}pi_hD_i+\sum_{i=1}^{n}\sum_{j=1}^{T_i}\sum_{h=1}^{m}\sum_{q=1}^{k}y_{ijh}x_{ijq}P_{hq}r_{ijq} \tag{6}$$

满足于：

$$\sum_{h=0}^{m}y_{ijh}=1,i=1,2,\cdots,n;j=1,2,\cdots,T_i \tag{7}$$

$$\sum_{i=1}^{n}\sum_{q=1}^{k}\sum_{j=1}^{T_i}z_{ijs}x_{ijq}CPU_q\leqslant CPU^*,s=0,1,\cdots,S \tag{8}$$

$$\sum_{i=1}^{n}\sum_{q=1}^{k}\sum_{j=1}^{T_i}z_{ijs}x_{ijq}Mem_q\leqslant Mem^*,s=0,1,\cdots,S \tag{9}$$

式(6)为目标函数，即总成本由3个部分组成：超过交付期限制的总惩罚、外包任务的数据传输成本和使用公有云提供的虚拟机实例的成本。式(7)用于保证一个任务只能分配到一个云上；式(8)和式(9)施加了一组约束，即在私有云上消耗的资源总量在任何时刻都不能超过相应的资源限制(即CPU^*和Mem^*)。

建立在一组二进制决策变量的基础上，式(6)－式(9)中的问题可以归结为一个NP-难的整数规划问题，该问题在混合云环境下以不同的运行时间调度独立的任务。

3 算法设计

3.1 和声搜索算法的基础

和声搜索算法(Harmony Search，HS)是一种新兴的智能优化算法，模拟了音乐家创作音乐的过程。该算法通过不断地创作新的和声并更新和声记忆库(Harmony Memory，HM)，来使目标函数值随着算法的迭代而收敛，以此完成优化目标，在本文中即为不断探索新的BoT应用排列序列并更新解集，使得总成本随着算法的迭代而降低。

假设一个需要优化的函数为$f(X)$，且$X=\{x_1,x_2,\cdots,x_n\}\in R^n$。那么，可以把$X$看作一个由$n$个成员组成的乐队，他们用不同的乐器演奏出来的音调$x_i$组成的和声对应$X=\{x_1,x_2,\cdots,x_n\}$，函数$f(X)$可以看作对这组和声的总体美学评价，成员们根据评价不断地调整自己演奏的x_i(即搜索过程)，使得和声评价更高。其对应关系如表1所列。

表1 全局最优化问题与和声搜索算法问题的对比

Table 1 Comparison between global optimization problem and harmony search algorithm

类比项	全局最优化问题	和声搜索算法
最佳状态	全局最优解	最美和声
评价方式	目标函数	美学评价
单位元素	函数自变量取值	乐器音调
运行过程	一次迭代	一次创作

和声搜索算法自提出以来，因其原理清晰、易于实现、全局收敛性强等优点而得到了广泛的应用。

3.2 高效的和声搜索算法

HS算法的全局收敛性、全局搜索能力较强，但是也存在一些问题，如算法后期的收敛速度较慢，同时算法的局部探索能力也存在不足。另外，传统的HS算法在一轮迭代后只产生一个和声，这就造成了效率的极大浪费。为了弥补以上不足，本文提出了高效的和声搜索算法(Efficient Harmony Search，EHS)。高效的和声搜索算法如算法1所示。

算法1 高效的和声搜索算法

输入：所有应用程序的任务

输出：使总成本最小的任务序列

1. 将HM初始化，并设定参数T_{max}，HMCR，PAR等
2. while(未达到迭代最大次数) do
3. for c←1 to HMS do
4. 产生随机数R1；
5. if(R1＜HMCR) then
6. 和声从HM中随机取一个，产生随机数R2；
7. if(R2＜PAR) then
8. 对从HM中选取的和声进行微调；
9. else新和声不做任何变动；
10. else每个音调在取值范围内随机取值；
11. 将新和声加入HM，并根据目标函数式(6)计算新和声的适应度F_{new}；
12. end for
13. 将HM中的和声根据适应度排序，并清除目标值最低的一半和声；
14. end while
15. 返回最优和声 // 即使总成本最小的任务序列

第1步，本文通过随机组合任务序列的方式来初始化和声记忆库，一个任务序列即为一条和声，然后设定各项参数。

在第3－12步的循环中，先通过产生0到1之间的随机数R1并与HMCR比较大小，来决定是否从HM中随机取和声，再通过随机数R2与PAR比较大小来决定是否微调，以此产生HMS条新的和声，并根据目标函数计算出每条新和声的适应度。

第8步，为避免算法陷入局部最优，导致早熟，和声微调

步骤必不可少。本文采用以一定概率随机交换和声片段的方式来完成微调，即随机交换任务序列中两个任务的次序。

第 13 步，新和声记忆库的产生与传统的和声搜索算法中每次替代掉一条最差的和声不同，本文汲取了遗传算法中优胜劣汰的思想，将和声记忆库中的和声按照适应度值从大到小排列，然后淘汰掉排在末尾的一半和声，再形成新的和声记忆库。采用这种精英保留的策略，可以大大地提升算法的收敛速度，同时可以避免在算法的迭代过程中遗漏掉最优解。

算法 1 的时间复杂度主要由两部分决定，一部分是第 3 步到第 12 步的循环产生新和声，时间复杂度为 $O(n)$，另一部分是第 13 步中通过比较和声的适应度大小，来对记忆库中的和声进行冒泡排序，该步骤在最好情况下的时间复杂度为 $O(n)$，最坏情况下的时间复杂度为 $O(n^2)$，平均情况下时间复杂度为 $O(n^2)$。综上，本文提出的高效的和声搜索算法的时间复杂度为 $O(n^2)$。

本文提出的高效的和声搜索算法的关键在于生成新和声的步骤，该步骤不仅能够继承当前最优美的和声，而且能挖掘出隐藏在和声记忆库中的全局优良音调。

高效的和声搜索算法的流程如图 2 所示。

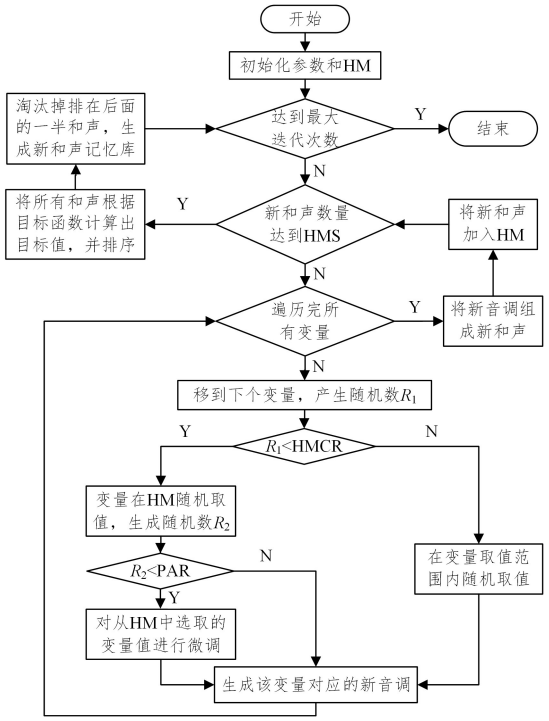

图 2　高效的和声搜索算法的流程图

Fig. 2　Flow chart of efficient harmony search algorithm

4　实验与分析

4.1　实验环境

本文实验在 Windows 10 系统下进行，CPU 型号为 Intel Core i7-6700HQ 2.60GHz，RAM 为 16GB，所使用的编程语言为 java 1.8。

4.2　实验设置

为了验证本文提出的 EHS 算法的有效性，我们构建了多组测试样例，以比较在不同问题规模下算法的性能。虚拟机实例的配置如表 2 所列。虚拟机的 CPU 和内存是决定虚拟机的两个关键因素，因此选择这两种资源类型作为虚拟机实例的参数。公有云虚拟机每小时的价格如表 3 所列。

表 2　虚拟机实例配置

Table 2　Virtual machine instance configuration

虚拟机类型	CPU/个	MEM/GB
t2. micro	1	1
t2. small	1	2
t2. medium	2	4
m3. medium	1	3.75
m3. large	2	7.5
m3. xlarge	4	15
m3. 2xlarge	8	30

表 3　公有云虚拟机每小时的价格

Table 3　Hourly price of public clouds virtual machine

（单位：$/h）

虚拟机类型	Prices(US)		
	EC2us-east	EC2us-west	EC2 eu-east
t2. micro	0.013	0.017	0.014
t2. small	0.026	0.034	0.028
t2. medium	0.052	0.068	0.056
m3. medium	0.070	0.077	0.077
m3. large	0.140	0.154	0.154
m3. xlarge	0.280	0.308	0.308
m3. 2xlarge	0.560	0.616	0.616

在这个实验中，我们设置了一个测试实例集来评估算法的性能。测试实例集由 3 组规模不同的问题组成，分别为小型问题、中型问题和大型问题。小型问题包含 5 个应用，中型问题包含 10 个应用，大型问题包含 20 个应用。每个应用的任务数量又分别设置为 5，10，20，因此产生了 5 * 5，5 * 10，5 * 20，10 * 5，10 * 10，10 * 20，20 * 5，20 * 10，20 * 20 这样 9 组不同规模的测试任务。每组测试任务集合又包含了 5 个耗时不同的测试实例，因此共有 45 个不同大小的测试实例。在我们的实验中，每个实例在不同的参数设置下都重复运行 5 次。每个应用所需的虚拟机类型是从表 3 中的 7 个虚拟机类型中随机选择的，每个任务的运行时间从 1 h 到 20 h 均匀分布。

4.3　参数测定

EHS 算法有两个重要的参数，即 *HMCR* 和 *PAR*。记忆库取值概率 *HMCR* 表示对之前优良和声的学习概率，取值范围为 0～1，它的值越大代表向优良和声学习的概率越大，值越小就代表全局搜索性能越强。微调概率 PAR 的值表示对和声进行微调的概率，同样在 0～1 之间取值，较小的 *PAR* 值可以增强算法的收敛性，而较大的 *PAR* 值可以增大算法的微调概率，防止算法陷入局部最优的情况。*HMCR* 和 *PAR* 的取值对算法的性能可能会有较大的影响，因此选择一组合适的算法参数是很有必要的。

本文做了多组实验来测定最合适的参数。首先设置 $HMCR\in\{0.1,0.2,0.3,0.4,0.5,0.6,0,7,0.8,0.9\}$，$PAR\in\{0.1,0.2,0.3,0.4,0.5\}$，因此对于本文算法而言，共有 45 种参数组合，在 9 组不同规模的测试任务上评估参数 *HMCR* 和 *PAR*，本文的参数测试实验的样本容量为 2025(45 * 9 * 5)。

为了评估本文算法，定义相对误差(RE)为：

$$RE=\left(\sum_{r=1}^{R}\frac{s_{\text{best}}-s_r}{s_{\text{best}}}\right)/R*100\% \tag{10}$$

其中，对于每一个实例而言，s_r 表示第 r 次实验的算法得出的目标函数值，s_{best} 表示在 R 次实验中目标值最优的一个。由于对同一个测试样例而言，s_{best} 是相同的，因此可以用 RE 来评估不同算法的有效性。显然，RE 值越小，算法的性能就越好。

图 3 和图 4 分别给出了 HMCR 和 PAR 对算法性能的影响，图中上面的曲线都表示 HS 算法的 RE，下面的曲线都表示 EHS 算法的 RE。经测定，随着 $HMCR$ 增大，HS 算法和 EHS 算法的 RE 都逐渐增大，即性能都逐渐下降；而 PAR 的取值对于两种算法的性能影响都不大。综合对比 $HMCR$ 和 PAR 的取值后发现，当 $HMCR=0.1$，$PAR=0.5$ 时，算法的综合性能最好。因此，在之后的实验中，都设置 $HMCR=0.1$，$PAR=0.5$。

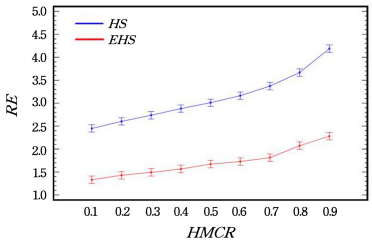

图 3 HMCR 对算法性能的影响

Fig. 3 Influence of HMCR on algorithm performance

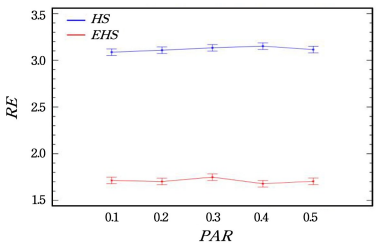

图 4 PAR 对算法性能的影响

Fig. 4 Influence of PAR on algorithm performance

4.4 实验结果

在下面的实验中，用 EHS 算法、HS 算法和当前解决此类问题最好的粒子群优化算法(PSO)[21]在同样的环境下运行相同的测试样例，以比较它们的性能。

经仿真实验运行并测算，我们得到了 EHS 算法、HS 算法和 PSO 算法的 RE 值。如表 4 所列，EHS 算法的 RE 最小值为 1.67，最大值为 1.74，平均值为 1.70；HS 算法的 RE 最小值为 3.08，最大值为 3.15，平均值为 3.11；PSO 算法的 RE 最小值为 2.24，最大值为 2.67，平均值为 2.46。

表 4 EHS 算法、HS 算法和 PSO 算法的 RE 值比较

Table 4 Comparison of RE values between EHS, HS and PSO

算法名称	RE		
	最小值	最大值	平均值
EHS	1.67	1.74	1.70
HS	3.08	3.15	3.11
PSO	2.24	2.67	2.46

图 5 和图 6 分别给出了 HS 算法与 EHS 算法以及 EHS 算法与 PSO 算法的性能比较结果，显示了各算法在具有 95% 置信水平的 LSD 区间的平均 RE 值。

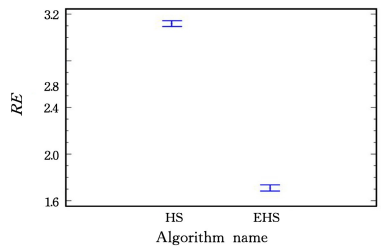

图 5 HS 算法与 EHS 算法的性能比较

Fig. 5 Performance comparison between HS and EHS

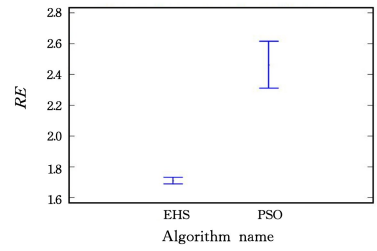

图 6 EHS 算法与 PSO 算法的性能比较

Fig. 6 Performance comparison between EHS and PSO

由表 4 和图 5 可以看出，相比 HS 算法，EHS 算法的 RE 值要小很多，这表明 EHS 算法有更好的性能，可以看出本文提出的改进方案对于算法性能的提升效果显著，验证了高效的和声搜索算法的有效性。

由表 4 和图 6 可以看出，与 PSO 算法相比，EHS 算法的 RE 值也要小很多，并且 RE 值更收敛，这表明了 EHS 算法相比 PSO 算法也有较好的有效性和稳定性。

结束语 本文以最大化收益为目标，研究了混合云环境下具有交付期约束的 BoT 应用程序的调度问题。本文在基本的和声搜索算法的基础上，结合遗传算法中优胜劣汰的思想，提出了一种高效的和声搜索算法。该算法优化了更新和声记忆库的方式和生成新和声的步骤，可以在同等环境下得到更优的 BoT 应用调度方案。本文通过详细具体的仿真实验来评估该算法的性能。实验结果表明，相比传统的和声搜索算法和当前解决此类问题最好的粒子群优化算法，本文提出的高效的和声搜索算法的性能都有明显的提升，并且有良好的稳定性。

本文关注具有用户指定的服务质量要求的计算密集型 BoT 应用程序。下一步将研究数据密集型和计算数据双密集型的 BoT 应用程序的调度。

参 考 文 献

[1] HU M, VEERAVALLI B. Requirement-Aware Scheduling of Bag-of-Tasks Applicationson Grids with Dynamic Resilience[J]. IEEE Transactions on Computers, 2012, 62(10): 2108-2114.

[2] TERZOPOULOS G, KARATZA H D. Bag-of-Task Scheduling on Power-Aware Clusters Using a DVFS-Based Mechanism [C]// 2014 IEEE International Parallel & Distributed Processing Symposium Workshops. IEEE, 2014: 833-840.

[3] THAI L, VARGHESE B, BARKER A. A Survey and Taxonomy of Resource Optimisation for Executing Bag-of-Task Applications on Public Clouds[J]. Future Generation Computer Systems, 2018, 82: 1-11.

[4] XIE G，GANG Z，YAN L，et al. Fast Functional Safety Verification for Distributed Automotive Applications during Early Design Phase[J]. IEEE Transactions on Industrial Electronics，2017，65(5)：4378-4391.

[5] LI J，XIE G，LI K，et al. Enhanced Parallel Application Scheduling Algorithm with Energy Consumption Constraint in Heterogeneous Distributed Systems[J/OL]. Journal of Circuits，Systems and Computers，2019，28(11). https://www.worldscientific.com/doi/abs/10.1142/S0218126619501901.

[6] XIE G，HUANG J，LI Y，et al. System-Level Energy-Aware Design Methodology Towards End-To-End Response Time Optimization[J/OL]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems，2019. https://ieeexplore.ieee.org/abstract/document/8732346.

[7] ZHOU J L，HU X S，MA Y，et al. Improving Availability of Multicore Real-Time Systems Suffering Both Permanent and Transient Faults[J]. IEEE Transactions on Computers，2019，68(12)：1785-1801.

[8] LEI M，KRITIKAKOU A，SENTIEYS O. Energy-Quality-Time Optimized Task Mapping on DVFS-enabled Multicores[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems，2018，37(11)：2428-2439.

[9] MO L，KRITIKAKOU A，SENTIEYS O. Controllable QoS for Imprecise Computation Tasks on DVFS Multicores with Time and Energy Constraints[J]. IEEE Journal on Emerging and Selected Topics in Circuits and Systems，2018，8(4)：708-721.

[10] WANG J，QIU M，GUO B. Enabling real-time information service on telehealth system over cloud-based big data platform[J]. Journal of Systems Architecture，2017，72：69-79.

[11] GARCÍA-VALLS M，DUBEY A，BOTTI V. Introducing the new paradigm of Social Dispersed Computing：Applications，Technologies and Challenges[J]. Journal of Systems Architecture，2018，91：83-102.

[12] WU T，GU H，ZHOU J，et al. Soft Error-Aware Energy-Efficient Task Scheduling for Workflow Applications in DVFS-Enabled Cloud[J]. Journal of Systems Architecture，2018，84：12-27.

[13] ZHOU X，ZHANG G，SUN J，et al. Minimizing cost and makespan for workflow scheduling in cloud using fuzzy dominance sort based HEFT[J]. Future Generation Computer Systems，2019，93：278-289.

[14] ZHOU J，JIN S，ZHOU X，et al. Resource Management for Improving Soft-Error and Lifetime Reliability of Real-Time MPSoCs[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems，2018，38(12)：2215-2228.

[15] TERZOPOULOS G，KARATZA H D. Bag-of-Task Scheduling on Power-Aware Clusters Using a DVFS-Based Mechanism[C]// 2014 IEEE International Parallel &Distributed Processing Symposium Workshops. IEEE，2014：833-840.

[16] ZHANG Y，SUN J，ZHU J. An Effective Heuristic for Due-Date-Constrained Bag-of-Tasks Scheduling Problem for Total Cost Minimization on Hybrid Clouds[C]// 2016 IEEE International Conference on Progress in Informatics and Computing. 2016：479-486.

[17] BOSSCHE R，VANMECHELEN K，BROECKHOVE J. Cost-Efficient Scheduling Heuristics for Deadline Constrained Workloads on Hybrid Clouds[C]// IEEE Third International Conference on Cloud Computing Technology & Science. IEEE，2011：320-327.

[18] WANG B，SONG Y，SUN Y，et al. Managing Deadline-Constrained Bag-of-Tasks Jobs on Hybrid Clouds with Closest Deadline First Scheduling[J]. KSII Transactions on Internet & Information Systems，2016，10(7)：2952-2971.

[19] PELAEZ V，CAMPOS A，GARCIA D F，et al. Autonomic Scheduling of Deadline-constrained Bag of Tasks in Hybrid Clouds[C]// International Symposium on Performance Evaluation of Computer & Telecommunication Systems. IEEE，2016：1-8.

[20] ARDAGNA D，CASOLARI S，PANICUCCI B. Flexible Distributed Capacity Allocation and Load Redirect Algorithms for Cloud Systems[C]// IEEE International Conference on Cloud Computing. Washington DC，USA：IEEE，2011：163-170.

[21] ZHANG Y，SUN J. Novel Efficient Particle Wwarm Optimization Algorithms for Solving QoS-demanded Bag-of-tasks Scheduling Problems with Profit Maximization on Hybrid Clouds[J]. Concurrency and Computation：Practice and Experience，2017，29(21)：e4249.1-e4249.19.

[22] ZHANG Y，ZHOU J，SUN J. Scheduling Bag-of-Tasks Applications on Hybrid Clouds Under Due Date Constraints[J/OL]. Journal of Systems Architecture. https://www.sciencedirect.com/science/article/pii/S1383762119304618.

[23] STAVRINIDES G L，KARATZA H D. Performance Evaluation of a SaaS Cloud Under Different Levels of Workload Computational Demand Variability and Tardiness Bounds[J]. Simulation Modelling Practice and Theory，2019，91：1-12.

[24] ZHANG X S，YU D H，NIE X C，et al. Spatiotemporal crowdsourcing online task allocation based on predictive analysis[J]. Computer Engineering，2019，45(6)：67-74.

[25] WANG C X，YU D H，ZHANG W S，et al. Module Allocation Algorithm for Software Crowdsourcing Based on Core Degree Sorting[J]. Computer Engineering，2019，45(7)：66-70.

[26] XING H，CHEN R，TANG W J. Online Task Allocation Strategy for Spatial Crowdsourcing Based on Prediction Algorithm[J]. Computer Engineering，2020，46(9)：27-34，43.

YAN Lei，born in 1996，postgraduate. His main research interests include cloud computing，Web service and distributed computing system.

ZHANG Gong-xuan，born in 1961，Ph.D，professor，is a member of China Computer Federation. His main research interests include cloud computing，Web services and distributed system.

（责任编辑：喻藜）