

# 改进的猴群算法在云计算资源分配中的研究<sup>①</sup>

陈海涛

(中国食品药品检定研究院, 北京 100050)

**摘 要:** 如何能够更好的解决云计算资源分配问题一直都是研究的热点, 引入猴群算法, 针对猴群算法中出现的局部收敛速度快, 容易造成局部最优值的缺点, 首先在猴群算法中引入混沌算法和反向学习来初始化猴群的初始位置, 其次, 通过猴群算法中的爬, 望, 跳三个过程的改进使得改算法收敛精度提高. 通过经典函数测试后, 本文算法相比其他智能算法的性能有了明显的改进. Cloudsim 平台证明将本文算法运用到云计算资源分配中, 在任务完成时间, 资源消耗方面都有了很大的提高.

**关键词:** 猴群算法; 混沌算法; 反向学习

## Resource Allocation in Clouding Computing Based on Monkey Algorithm

CHEN Hai-Tao

(National Institutes for Food and Drug Control, Beijing 100050, China)

**Abstract:** It has always been the focus of research as how to solve the resource distribution problem in cloud computing, and aiming at the weakness emerging in the monkey algorithm, i.e. too quick convergence speed and being easy to fall into local optimal, etc. the monkey algorithm is introduced in this paper. First, chaos algorithm and inverse learning are introduced into monkey algorithm to initialize the initial position of monkey swarm. Second, the improvement of climbing, watching and jumping in monkey algorithm has improved the convergence precision of this algorithm. After the test of classic function, algorithm in this paper has improved noticeably in performance compared with other intelligent algorithms. Cloudsim platform has proved that by applying algorithm in this paper to cloud computing, resource consumption has been greatly improved during completing tasks.

**Key words:** monkey algorithm; chaos algorithm; backward learning

云计算是近几年来互联网发展最快的一种协同、共享的计算方式, 通过云计算可以实现软硬资源共享, 避免分布式网络信息冗余等缺点<sup>[1]</sup>. 但由于资源共享会产生一系列的问题, 比如多个云端用户在向服务器申请任务的时候, 可能出现一台服务器受理所有的请求从而导致该台服务器陷入瘫痪, 而同时其他服务器都处于空闲的状态, 造成这个情况的主要原因是因为系统的资源分配负载不均衡导致出现的崩溃的情况. 针对云计算资源分配不均匀的问题, 国内外学者将人工智能算法引入到资源分配中, 在一定程度上起到了非常好的效果. 文献[2]提出一种基于蚁群优化(Ant Colony Optimization)的计算资源分配算法, 该算法利用

蚁群优化算法得到一组最优的计算资源. 仿真实验证明分配算法更短的响应时间和更好的运行质量, 因而更加适合于云环境. 文献[3]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(IGA), 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环境资源调度, 但缺点是消耗资源和时间太大. 文献[4]提出将遗传算法中的量子位的二进制编码转换为实数编码, 并使用旋转策略和变异算子保证算法的收敛性, 仿真实验表明此算法能取得更小的最小服务成本. 文献[5]提出一种基于文化粒子群算法的云计算资源调度算法, 仿真结果表明本文算法获得更短的任务完成时间, 使资源节点负载更加均衡, 尤其对于大规模任务的

<sup>①</sup> 收稿时间:2014-12-16;收到修改稿时间:2015-01-29

云计算资源调度优势更加明显.文献[6]在云计算资源分配下提出了一种改进型的人工蜂群算法.该算法通过对邻近因子的修改,使得改进后的算法能够有效提高局部搜索能力.改进后的算法可以有效的减少任务处理请求的平均完成时间,从而提高了云计算下的任务处理的效率.

针对云计算的资源分配问题,本文在云计算中引入一种改进的猴群算法,首先本文对猴群算法容易造成收敛速度快,容易陷入局部最优的情况进行改进,引入混沌算法和反向学习算法,从而使得猴群在初始阶段能够合理的分配,其次针对猴群算法中爬,望,跳三个过程进行改进.通过经典测试函数表明本文算法性能有了明显的提高,Cloudsim 仿真平台证明本文算法能够有效的提高云计算中的资源调度效率,缩短资源分配时间,对于云计算的资源分配起到一定的效果.

## 1 云计算资源的QoS服务

资源调度是云计算中一个关键性的问题,它决定了云计算中用户执行的效率,系统使用的资源数目以及云计算系统的整体性能,为了更好来判断云计算中资源调度的是否合理,采用服务质量(Quality of Service,QoS)来进行衡量有关用户使用云计算服务是否满意的重要因素.云计算为各类用户提供服务,从而满足不同用户的不同需求,由于云计算自身的用户具有的多样性,因此对于云计算的作业调度和资源分配都提出了更好的要求,从云计算的 QoS 的服务要求出发将云计算中的任务调度按照如下进行分类处理,可以更加准确的及时将任务分配到最合适的资源中,通常考虑如下几个要求:

1)网络带宽:云计算中的网络带宽由于受到来自不同地段的条件要求,带宽是保证云计算资源调度的最基本的要求.

2)任务完成时间:由于云计算中涉及的用户类型不同,可能存在一些实时用户需要在尽可能短的时间内及时完成任务的调度.

3)任务所需费用:云计算是需要按时间付费的,特别是处于云端的用户希望能够采用相对合理的价格来进行使用云资源

4)系统的可靠性:由于运行许多的复杂任务,需要云计算中心提供相对稳定可靠的服务

通过以上说明可以发现在云计算资源调度中 QoS 服务主要关心的是效率和性能的基本要素,就云计算本身而言,应该是考虑资源调度的质量和算法的效率来作为整个资源调度的评价的依据,特别是在整个目前资源质量无法统一的情况下,调度算法的效率就作为资源调度的关键.

## 2 猴群算法

猴群算法(monkey algorithm,简称 MA)<sup>[8]</sup>是一种解决大规模优化问题的智能算法,其主要思想是模仿猴群爬山行为,模拟猴子在爬山过程中的爬,望,跳等动作实现的最优解的搜索.在求解的过程中,使用  $N$  表示求解空间的维度,使用  $M$  表示猴群种群的大小,则第  $i$  只猴子的空间位置表示如下:  $X_i = (x_{i1}, x_{i2}, \dots, x_{in})$   $i \in [1, M]$ , 其中  $x_{ij}$  表示每一只猴子在一维空间中的实际位置,每一只猴子的实际位置代表了优化问题的一个决策值.

### 2.1 爬过程

爬过程是一个不断改善逐步优化的目标函数数值的过程.借鉴 SPSA 想法<sup>[9]</sup>,设计过程如下:

1) 随机生成向量  $\Delta x_i = (\Delta x_{i1}, \Delta x_{i2}, \dots, \Delta x_{in})$ , 其中,  $\alpha$  为猴子爬的步长,设定  $\Delta x_i$  以相同的概率 0.5 来进行取值  $-\alpha$  或者  $\alpha$ ,  $\alpha$  取值大小关系到优化问题最优解的精度问题.

2) 计算  $f'_{ij}(x_i) = \frac{f(x_i + 2\Delta x_i) - f(x_i)}{2\Delta x_{ij}}$  ( $j=1, 2, \dots, n$ ) 其

中  $f'_{ij}(x_i)$  计算的值组成如下  $\{f'_{i1}(x_i), f'_{i2}(x_i), \dots, f'_{in}(x_i)\}$  分别是每一个目标函数伪梯度.

3)  $y_i = x_{ij} + \alpha \cdot \text{sign}(f'_{ij}(x_i))$ ,  $j=1, 2, \dots, n$  且  $y = \{y_1, y_2, \dots, y_n\}$

4) 当  $y$  满足约束函数的要求也就是具有可行解的时候,  $y \rightarrow x_i$ , 否则保持  $x_i$  的值不变.

重复以上步骤(1)到步骤(4),直到达到最大爬次数或者达到完成的次数为止.

### 2.2 望过程

通过前面的爬的过程,每只猴子都能达到各自的山峰顶出的时候,也就是说暂时获得了局部最优解,然后,它观察周围附近的地区是否存在比当前更好的山峰,如果存在就会想更高的位置跳过去,因此寻

找更高的位置是一个望的过程. 在猴群算法中定义了一个参数  $\beta$  叫做猴群的视野距离. 定义如下:

- 1) 在视野范围内  $(x_{ij} - \beta, x_{ij} + \beta)$  内随机选取相邻点, 并随机产生一个  $y_i$ , 且  $y = (y_1, y_2 \cdots y_n)$
- 2) 如果存在  $f(y) > f(x_i)$  且满足  $y$  时有效解的时候,  $y \rightarrow x_i$ , 否则扩到(1)中的范围, 直到寻找到合适的  $y$ .
- 3) 在新的  $y$  产生的时候, 重复做这个动作.

### 2.3 跳过程

跳的过程主要是为了在搜索过程中将当前区域转移到新的区域的过程中, 选择所有的猴子的重心位置作为支点, 每只猴子会沿着当前的位置指向支点的方向翻到各自新的搜索区域, 对于第  $i$  只猴子而言跳的过程如下:

- 1) 从一个区间  $[a, b]$  中随机产生一个实数  $\theta$
- 2) 令  $y_j = x_{ij} + \theta(p_j - x_{ij})$ , 其中 
$$p_j = \frac{\sum_{l=1}^M (x_{lj} - x_{ij})}{M - 1}$$
,  $p_j$  的集合成为支点.

- 3) 如果  $y$  是可行的, 就将  $y \rightarrow x_i$ , 否则重复执行 1)、2)直到找到新的可行解.

## 3 改进的猴群算法在云计算中的描述

云计算资源分配和猴群算法的寻找食物具有一定的相似性, 表现在以下三点: 1)猴群算法中的猴群个体类似于云计算中的云端用户对系统资源的请求; 2)猴群算法中的食物类似于云计算的云端服务器的资源; 3)猴群算法寻找到最佳的食物过程就是云计算中寻找资源分配最优解的过程.

### 3.1 初始化位置改进

对于猴群算法中的猴群的初始位置没有特别的说明, 但是猴群个体的初始位置跟最终产生的最优解有着一定的关系, 如果猴群个体的初始化位置经过筛选和排列之后, 在一定程度上可以节省算法所需要找到最优解的时间. 本文针对初始化位置采用混沌和反向学习的方法来对猴群算法进行初始化.

由于混沌算法具有更好的遍历性和不重复性的特点, 因此, 本文采用混沌方法来进行设置初始位置, 本文采用经典的 Logistic 函数来产生混沌变量<sup>[7]</sup>, 其函数的为:

$$y^k = 4y^{k-1}(1 - y^{k-1}) \quad (1)$$

式(1)中  $y$  是混沌变量,  $k$  是迭代次数.

从式(1)中发现  $y^{k-1}$  的值是属于  $(0, 1)$  通过实验发现  $y^{(0)} \neq \{0, 0.25, 0.5, 0.75\}$  的时候,  $y^{k-1}$  的值可以访问整个值域的内容.

在值域范围  $[a, b]$  之间产生混沌变量转换到优化问题的定义域内变成如下

$$y^k = \frac{x^k - a}{b - a} \quad (2)$$

反向学习阶段是也是一种处理初始化的方法, 能够有效解决数值分布不均的情况.

```
for i = 1 to a do
  for j = 1 to b do
     $y'_{ij} = y_{\min,j} + (y_{\max,j} - x_{i,j})$ 
  endfor
endfor
```

结合式(2)和式(3), 从  $\{Y(M) \cup Y'(M)\}$  中选取适应度值最好的猴子个体序列作为初始种群.

### 3.2 爬过程的改进

从前述描述的爬的过程来看,  $\alpha$  值是非常关键, 当  $\alpha$  值越小的时候求解的精度就越高, 但需要花费 CPU 比较长的时间完成过程. 反之则花费 CPU 时间较短. 设置步长的次数与  $\alpha$  呈反比, 当  $\alpha$  为 0.01 的时候, 步长的次数为 100 次, 根据概率计算在每次进行迭代的时候需要计算 200 次, 依次类推, 当  $\alpha$  值逐渐减少的时候, 步长的次数越来越大, 所需要进行迭代的次数也越来越大. 为了解决这个问题, 本文通过定位步长的方法来进行平衡, 就是在每次迭代过程中逐渐减少步长的次数, 当  $\alpha$  值相对比较大的时候, 这样可以更好的进行全局最优值的查找, 当  $\alpha$  值出现比较小的时候, 则比较有利于寻找局部最优值. 设定  $\alpha$  值的范围为  $[\alpha_{\min}, \alpha_{\max}]$ , 算法中的最大迭代次数  $Max$ , 最小迭代次数为  $Min$ ,  $\lambda$  为目前的迭代次数, 因此  $\alpha^\lambda$  为迭代次数对应的步长, 按照如下公式设定:

$$\alpha^\lambda = e^{-\frac{\lambda}{Max+Min}} \bullet \alpha^{\lambda-1} \quad (4)$$

在进行开始迭代的过程中, 采用  $\alpha_{\max}$  作为初始步长的值, 伴随着迭代次数的不断增加,  $\lambda$  的值逐渐增大, 但是  $e^{-\frac{\lambda}{Max+Min}}$  的值逐渐减少, 这样就能保持步长  $\alpha$  值逐渐逼近  $\alpha_{\min}$ , 从而可以避免算法无限制的去搜索还无法收敛.

### 3.3 望过程的改进

在经过爬的过程之后, 猴子个体  $i$  都达到了各自所在的山峰的山顶. 参数  $\beta$  决定了猴群的视野也就是能看到的最远的距离, 当  $\beta$  的值越小的时候, 就容易陷入局部最优, 当  $\beta$  的值越大的时候, 算法容易导致收敛速度变慢. 为了平衡这种情况, 本文对  $\beta$  进行如下设置

$$\beta^\lambda = \frac{\lambda}{Max} \cdot \beta^{\lambda-1} \quad (5)$$

其中,  $\lambda$  表示迭代次数,  $Max$  为最大次数, 因此望过程的视野范围改为  $(x_{ij} - \beta^\lambda, x_{ij} + \beta^\lambda)$ , 伴随着迭代次数的增多,  $\beta$  的值逐渐达到最大值, 从而有效减少快速陷入局部最优的几率, 同时能够保持算法的收敛速度逐步达到平衡.

### 3.4 跳过程的改进

在跳的过程中, 主要是将当前区域转移到新的区域中, 在猴群算法中猴子的向着支点的方向进入各自新的区域, 对于  $x_{ij}$  按照如下公式进行更新:

$$x_{ij}^\lambda = p_j + \theta(p_j - \beta x_{ij}^{\lambda-1}) \quad (6)$$

利用在爬过程的参数  $\beta$  来更新在每一次迭代  $\lambda$  中的猴子的位置向量, 通过  $p_j$  作为跳过程中的支点, 能够避免猴子选择跳过程的距离不会随意太大, 获得猴群在翻过程中的翻阅后的位置.

### 3.5 算法步骤

步骤 1: 设定猴群算法的参数, 种群大小为  $M$ , 设置爬步长, 视野范围及其其他的参数, 设置云计算的资源数目, 服务器的数目, 云端用户访问数目, 将猴群个体最优与云端用户的获取最优资源进行对应.

步骤 2: 在初始化后, 将猴群个体通过(2)和(3)的处理之后, 来获得初始化的位置.

步骤 3: 在爬的过程中根据改进的伪梯度来计算猴群个体的位置. 同时搜索更新位置, 并更新猴群位置到最优位置.

步骤 4: 对猴群算法进行望处理, 采用(5)搜索到新的猴群个体的位置.

步骤 5: 对猴群算法进行跳处理, 采用公式(6)获得跳过程之后的猴群个体新的位置.

步骤 6: 通过三个过程之后是否发现猴群个体找到了最佳的食物, 是否达到了最优解, 如果达到, 则记录位置, 退出算法, 如果不满足, 则算法结束否

则转到步骤 3.

步骤 7: 输出找到食物的最优猴群的个体位置, 从而得到对应的最优位置的向量, 也就是云计算中资源分配的最佳结果.

## 4 仿真实验

本实验首先验证改进后的猴群算法的性能, 然后通过云计算分配模型中的运用来获得资源调度的性能. 本文一方面测试改进后的本文算法的性能, 另一个方面来测试来云计算平台下的调度资源的效果. 本实验采用运行环境为处理器为 AMD(Athlon)III3.0 主频, 内存为 4GDDR3, WindowsXP 系统,

### 4.1 算法性能检验

在算法中采用 Matlab2012, 设置种群规模为, 每次优化的过程中猴子的个数为 3, 步长的值为 [0.001,0.01], 为 0.5, 爬的次数为 1000 次, 爬的次数为 1000. PSO 算法中设置种群的规模大小为 500, GA 算法设置种群设置为 1000, 交叉概率为 0.9, 变异概率为 0.02. 本文选取了 4 个经典函数, 采用如表 1 所示的测试函数. 表 2 表示采用本文算法与其他的两种智能算法通过 4 个函数的比较结果.

表 1 测试函数

函数	搜索空间	维数
$f_1(x) = \sum_{i=1}^n x_i^4 + random(0,1)$	[-1.28,1.28]	30
$f_2(x) = \sum_{i=1}^n (x_i^2 - 10 \cos(2\pi x_i) + 10)$	[-5.12,5.12]	30
$f_3(x) = \frac{1}{4000} \sum_{i=1}^n x_i^2 - \prod_{i=1}^n \cos(\frac{x_i}{\sqrt{i}}) + 1$	[-600,600]	30
$f_4(x) = -(\sum_{i=1}^n  x_i ) \exp(-\sum_{i=1}^n x_i^2)$	[-6,6]	30

表 2 本文算法与 MA 算法, PSO 算法和 GA 算法比较

测试函数	算法	最好值	最差值	方差值
$f_1$ 函数	GA	1.24e-2	2.24e-2	8.93e-3
	PSO	3.12e-3	4.19e-3	6.17e-4
	MA	6.15e-3	2.37e-2	5.72e-3
	本文算法	7.17e-5	9.35e-5	8.18e-5
$f_2$ 函数	GA	2.02	1.05	0.74
	PSO	37.25	78.09	17.29
	MA	3.01	12.97	8.19
	本文算法	1.14	5.19	3.92
$f_3$ 函数	GA	20.71	120.52	100.71
	PSO	1.19	4.29	2.18
	MA	1.05e-2	7.37e-2	4.92e-3
	本文算法	0.37e-2	4.18e-2	2.71e-4
$f_4$ 函数	GA	3.56e-10	4.32e-12	3.97e-13
	PSO	1.76e-11	8.26e-7	2.23e-7
	MA	3.65e-14	7.51e-13	9.14e-5
	本文算法	7.41e-16	9.13e-15	3.99e-12

从表 2 中可以发现, 与 GA, PSO, MA 相比, 本文算法通过 4 个经典测试函数比较后, 搜索最优解的性能最强, 从低维或者高维的空间来看, 求解的最优值都是最佳的. 测试函数的比较如图 1 到图 4 所示.

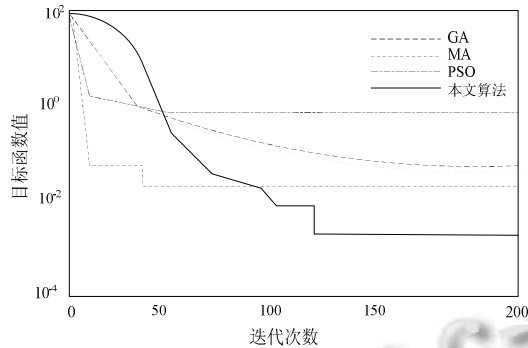


图 1 四种算法在  $f_1$  函数测试结果

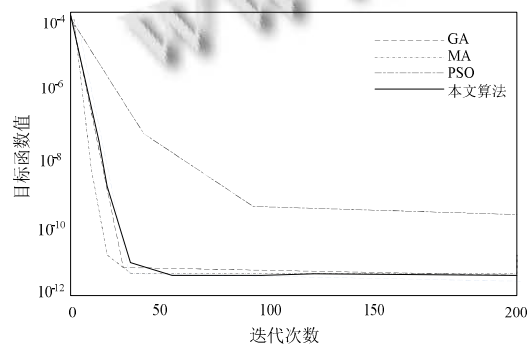


图 2 四种算法在  $f_2$  函数测试结果

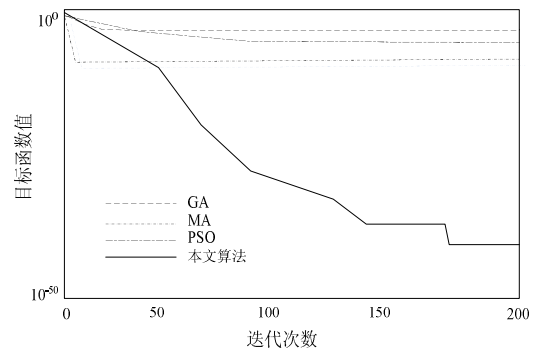


图 3 四种算法在  $f_3$  函数测试结果

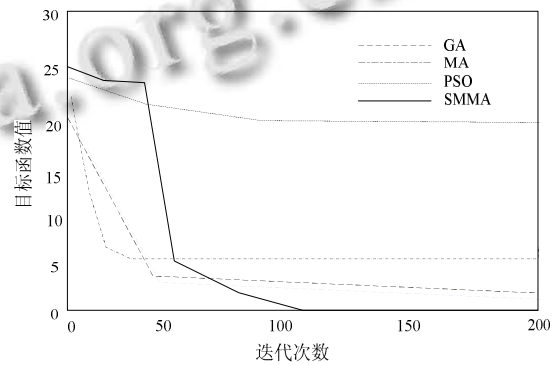


图 4 四种算法在  $f_4$  函数测试结果

#### 4.2 云计算资源调度

在云计算的资源算法实验中, 采用 Cloudsim 平台对资源进行分配调度. 从以下三个方面来进行.

##### 1) 与基本猴群算法比较

设定虚拟任务为 800 个, 虚拟节点为 100 个, 通过比较两种算法下的能量消耗可以发现, 本文的算法在开始的能量消耗比较大, 后期慢慢趋于平缓, 主要是因为采用了初始位置采用了混沌算法和反向学习, 消除了算法初期震荡大的情况, 本文的算法消耗低于基本猴群算法的消耗. 如图 5 所示.

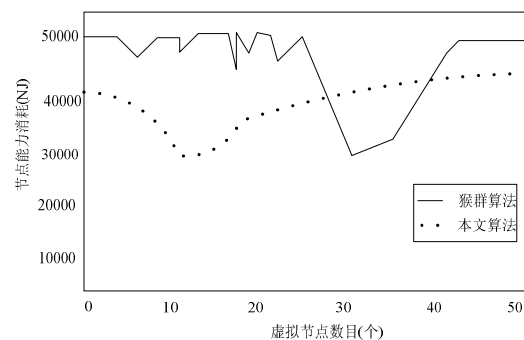


图 5 2 种资源负载算法能量消耗比较

2) 与其他智能算法进行比较

设定虚拟任务为 800 个, 虚拟节点为 70 个, 设置迭代次数为 300, 将本文的算法和文献[2], 文献[3]和文献[6]的算法在云计算模型中不同任务数和虚拟节点下进行比较。

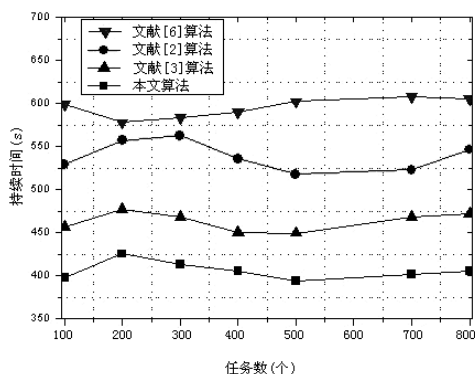


图 6 4 种算法的任务完成时间比较

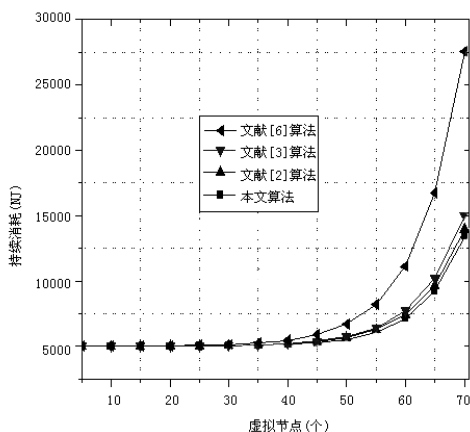


图 7 4 种算法的能量消耗比较

从图 6 中发现, 本文算法在资源负载方面消耗时间是低于其他三种参考文献算法, 同时伴随着任务数目不断的增多, 消耗的时间震荡幅度是最小的, 说明本文的算法能够有效的平衡资源。从图 7 中发现经过改进后的本文算法在云计算环境中的网络消耗持续方面优于其他的三种参考文献算法。从而能够更好地满足云计算中的资源调度的要求。

5 结语

本文根据云计算资源算法分配的特点引入猴群算法, 针对猴群算法中出现的局部收敛速度快, 容易造成局部最优值的问题, 首先在猴群算法中引入混沌算法和反向学习来初始化猴群的初始位置, 其次, 通过猴群算法中的爬, 望, 跳三个过程进行改进, 使得改进后的算法收敛精度提高, 仿真实验表明本文算法能够有效满足云计算资源分配, 同时在任务完成, 资源能量消耗方面都有了很大的提高, 为云计算的资源分配提供了一种参考。

参考文献

- 林伟伟, 齐德昱. 云计算资源调度研究综述. 计算机科学, 2012, 39(10): 1-5.
- 华夏渝, 郑骏, 胡文心. 基于云计算环境的蚁群优化计算资源分配算法. 华东师范大学学报(自然科学版), 2010, 32(1): 127-134.
- 刘愉, 赵志文, 李小兰, 等. 云计算环境中优化遗传算法的资源调度策略. 北京师范大学学报(自然科学版), 2012, 48(4): 378-383.
- 刘卫宁, 靳洪兵, 刘波. 基于改进量子遗传算法的云计算资源调度. 计算机应用, 2013, 33(8): 2151-2153.
- 孟令玺, 李洪亮. 基于 CA-PSO 算法的云计算资源调度策略. 计算机仿真, 2013, 30(10): 406-410.
- 黄华. 一种改进型的人工蜂群算法在云计算的资源分配中的研究. 科技通报, 2013, 29(5): 142-146.
- 郝士鹏. 混沌猴群算法及其应用[硕士学位论文]. 天津: 天津大学, 2010.
- Zhao RQ, Tang WS. Monkey algorithm for global numerical optimization. Journal of Uncertain Systems. 2008, 2(3): 164-175.
- Spall J. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. IEEE Trans. on Automatic Control, 1992, 37(3): 332-341.