云计算中的基于粒子群算法和差分遗传算法的资源调度①

陈海涛

(中国食品药品检定研究院, 北京 100050)

摘 要: 云计算资源调度一直以来都是研究的热点, 本文在云计算中引入粒子群算法, 针对该算法局部收敛速度快, 容易陷入局部最优值的缺点. 本文提出了两个改进: 一个是在粒子群种群寻找最优解中引入差分遗传算法, 既可以发挥粒子群全局搜索快的优点, 又可以发挥差分遗传算法局部搜索效率高的优点, 将两种算法优点进行结合弥补粒子群算法不足; 另一个是引入惩罚函数避免了粒子向无效的空间移动, 节约了移动的成本. Cloudsim 平台说明本文算法能够有效满足云计算资源分配, 同时在任务完成时间, 成本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.

关键词: 云计算; 粒子群算法; 差分遗传算法; 惩罚函数

Resource Scheduling Based on Particle Swarm and Differential Genetic Algorithm in Cloud Computing

CHEN Hai-Tao

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

Abstract: Resource scheduling in cloud computing has long been the focus of research. In this paper, particle swarm algorithm is introduced into the cloud computing and aiming at the shortcomings of this algorithm like fast local convergence and being easy to fall into local optimal value. Two improvements are made in this paper: one is to introduce differential genetic algorithm into particle swarm algorithm while finding the optimal solution, which can not only give play to particle swarm's advantage of quick global searching speed, but also give play to differential genetic algorithm's advantage of efficient local researching while combining the advantages of these two algorithms and making up for the deficiency of particle swarm algorithm. Another is to introduce penalty function so as to avoid particles moving towards void space and save costs of moving. Cloudsim platform shows that algorithm in this paper can effectively meet resource scheduling in cloud computing while having great improvement in reducing time of completing the task as well as consumption of costs so as to provide a reference for resource scheduling in cloud computing.

Key words: cloud computing; particle swarm algorithm; differential genetic algorithm; penalty function

0 引言

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[1]. 但由于资源共享会产生一系列的问题, 主要是因为系统的资源分配不均衡导致出现的情况. 因此在云计算的环境中, 需要更好的对云计算资源进行合理的分配才能尽量避免这样情况的发生. 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. 仿真实验证明该算法具有更短的

① 收稿时间:2015-02-10;收到修改稿时间:2015-04-17

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] 提出通过调节作业权值以及最小资源数目来进行对资源控制, 起到了一定的效果.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, 仿真实验平台表明此算法能取得更小的最小服务成本. 文献[5]将粒子群算法引入到云计算环境中进行多目标的优化, 结果表明可以优化处理和传输时间以及相应的传输费用. 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. 文献[7–8]分别将改进的遗传算法和粒子群算法运用到资源分配中, 起到了一定的效果.

本文首先描述了云计算的资源调度模型, 在云计算中引入粒子群算法, 针对该算法局部收敛速度快, 容易造成局部最优值的问题. 将粒子群种群寻找最优解中中加入差分遗传算法, 使得寻找过程中既能够发挥粒子群全局搜索快的优点, 同时也能够发挥差分遗传算法局部搜索效率高的两者优点, 同时引入惩罚函数使得粒子移动的效率提高, 避免了不必要时间和空间的移动. 仿真实验表明本文算法能够有效满足云计算资源分配, 在任务完成时间和任务成本消耗方面具有一定优势.

1 云计算资源调度描述

目前, 云计算中主要采用的是 Map/Reduce 模型, 将一个比较大的任务分解为许多个小的子任务, 然后分配给若干个虚拟资源节点并执行并返回结果, 本文从时间–成本的角度出发, 考虑子任务相互独立的情况. 通过将各个子任务进行合理的资源分配, 从而使得完成总任务的时间和成本达到最小.

云计算中的任务调度可以描述为将 n 个相互独立的子任务分配给 m 个资源($m<n$), 其中任务表示为 $Task=\{task_1,task_2,\cdots task_n\}$, $task_i$ 表示第 i 个子任务, 资源数目为 $\mathrm{Re}source=\{resource_1,resource_2,\cdots resource_n\}$, $resource_i$ 表示第 i 个资源, 因此每一个子任务只能在一个虚拟资源上执行, 任务 $Task$ 和 $\mathrm{Re}source$ 之间的关系适用矩阵来进行表示:

$$X=\begin{pmatrix} x_{11} & \cdots & x_{1n} \\ \vdots & \ddots & \vdots \\ x_{m1} & \cdots & x_{mn} \end{pmatrix} \tag{1}$$

公式(1)中 x_{ij} 表示子任务 i 与第 j 个资源的关系, 其值为 0 或者 1, 当值为 0 的时候, 表示子任务 i 没有占据第 j 个资源, 反之则占据. ETC_{ij} 表示子任务 i 在第 j 个资源上的理论上所需要的时间, 因此 ETC 矩阵表示如下:

$$ETC=\begin{pmatrix} ETC_{11} & \cdots & ETC_{1n} \\ \vdots & \ddots & \vdots \\ ETC_{m1} & \cdots & ETC_{mn} \end{pmatrix} \tag{2}$$

同理, 可以使用 RCU_{ij} 来表示子任务 i 在第 j 个资源上的理论上所需要的执行成本, 因此, RCU 矩阵表示如下:

$$RCU=\begin{pmatrix} RCU_{11} & \cdots & RCU_{1n} \\ \vdots & \ddots & \vdots \\ RCU_{m1} & \cdots & RCU_{mn} \end{pmatrix} \tag{3}$$

根据 ETC 矩阵和 RCU 矩阵可以得到各个资源节点在执行完所分配的子任务花费的时间和成本.

$$sumTime(i)=\max\sum_{i=1}^{resouce}\sum_{j=1}^{n}T(i,j)\times ETC(i,j) \tag{4}$$

$$sumCost(i,j)=\sum_{i=1}^{resource}\sum_{j=1}^{n}resource(i,j)\times RCU(i,j) \tag{5}$$

式中, $T(i,j)$ 表示子任务 i 中的第 j 个子任务的所需要时间, $resource(i,j)$ 表示子任务 i 中的第 j 个资源的所需要成本, 公式(4)和公式(5)分别表示任务完成的总时间和消耗的总成本. 云计算的资源调度的最终目的就是获得最大的任务执行时间和最小的消耗总成本.

2 基本算法

2.1 粒子群算法

基本粒子群算法(PSO)的思想是将空间中的个体比作是一个没有质量和体积的粒子, 通过粒子的速度和位置的不断变换来寻找粒子的最优解. 设群体中第 i 个粒子为 $X_i=\{x_{i1},x_{i2},....x_{im}\}$, 它经历过的位置为 $p_i=\{p_{i1},p_{i2},....p_{im}\}$, 其中, 单个粒子的最佳的位置为 p_{best}, 当前所组成的群体中的所有粒子经历过的最佳位置为 p_{gbest} , 粒子 i 的速度用

$v_i=\{v_{i1},v_{i2},.....v_{in}\}$ 表示. 粒子 i 运动的速度公式和位置公式分别为:

$$v_{id}(t+1)=w\times v_{id}(t)+c_1\times rand()\times (p_{best}-x_{id}(t))+c_2\times rand()\times(p_{gbest}-x_{id}(t)) \quad (6)$$

$$x_{id}(t+1)=x_{id}(t)+v_{id}(t+1) \quad (7)$$

其中, w 表示惯性权重, c_1 和 c_2 为常数, $rand()$ 为随机数, 取值在(0, 1)之间.

2.2 差分遗传算法

差分算法采用了遗传算法中的使用实数编码的特点, 是一种具有在连续空间能随机搜索的优化算法, 针对群智能算法中的含有 NP 个初始种群, 每一个解 $Z_i=(z_{i1},z_{i2}\cdots z_{in})$ 是一个具有 n 维向量的解, 主要有变异, 交互和选择三个部分构成.

(1)变异操作: 在差分算法中, 本文主要选择公式(8)的变异方式.

$$y_{ij}=z_{best,j}+F\square(z_{r1}-z_{r2}) \quad (8)$$

式(8)中的 $z_{best,j}$ 是当前种群中的最好个体的第 j 维向量, F 为随机因子, 主要是用来控制差分向量的缩放程度, 设定值为[0,1]之间.

(2)交叉操作: 通过一定的概率选择, 将变异的中间个体 y_i 与父代个体 z_i 之间进行交叉, 得到新的个体

$$\kappa_{i,j}=\begin{cases}y_{i,j}, & t\in[0,1]\\ z_{i,j}, & otherwise\end{cases} \quad (9)$$

式(9)中可以保证在交叉过程出现一个 0 到 1 之间的随机整数, 能够保证 $\kappa_{i,j}$ 至少有一个分量来自 $y_{i,j}$.

(3)选择操作: 差分算法在选择个体的时候采用“贪婪”选择策略, 从而能够保证适应度最优的个体选择到下一代中, 通过变异与交叉操作后生产的新的个体 $\kappa_{i,j}$ 与上一代个体 $y_{i,j}$ 进行比较, 如果小于则保持 $y_{i,j}$ 不变, 否则直接进入下一代.

3 粒子群–差分遗传算法在云计算中的资源分配

3.1 粒子的编码与云计算中任务的对应

根据云计算的特点, 本文采用资源–任务的间接编码方式, 即对每一个云计算中的子任务所占用的资源进行编码, 同时每一个子任务的数量决定了编码的长度, 这样将编码与云计算中的任务分配对应起来.

当任务的数目为 3, 云计算中的资源数目为 4, 将其中的任务 1 划分为 3 个子任务, 将任务 2 划分为 3 个任务, 将任务 3 划分为 4 个子任务, 划分完成后的总任务达到 10 个, 存在如下的一种可能任务调度方案.

表 1　粒子编码

任务序号	1	2	3
子任务序号	1,2,3	4,5,6	7,8,9,10
资源序号	4,1,2	3,2,1	4,2,1,3

表 2　粒子解码

资源序号	1	2	3	4
子任务序号	2,6,9	3,5,8	4,10	1,7

从表 1 中得到粒子的编码序列为(4,1,2,3,2,1,4,2,1,3), 因此对粒子的解码得到的结果如表 2 所示, 因此可以了解子任务{2,6,9}分配到了 1 号资源, 子任务{3,5,8}分配到了 2 号资源上, 子任务{4,10}分配到了3 号资源上, 子任务{1,7}分配到4 号资源上.

3.2 粒子群–差分遗传算法构成

由于粒子群算法与差分遗传算法在产生个体方式不同, 因此这两种算法在进行寻优的过程中产生的效果也不同, 综合前面的 2 种算法, 本文提出了一种差分的粒子群–遗传算法. 其思想是将初始化种群分为二组, 第一组采用粒子群算法, 第二组采用的是差分遗传算法. 在第一组中个体的寻找采用公式(6)和(7), 这样可以在粒子群种群中进行全局搜索, 在第二组中通过公式(8)进行更新, 变异个体来自当前的最优个体, 从而可以保证局部搜索能力的增强, 收敛速度快. 将第一组和第二组进行混合, 从而可以保证在维持种群多样性的时候加快算法的收敛, 使得算法能够在合理的角度下具有全局收敛效率.

3.3 惩罚函数

在公式(6)和(7)中由于粒子个体与周围领域个体之间存在一定的差异性,同时粒子群邻域的个体都没有进行初始化而进行的随机选择,同时没有考虑到当前粒子个体的速度和位置与相邻的粒子个体的速度和位置之间的差异, 从而导致粒子群算法在寻优进程中的粒子最优的速度和最优位置的控制能力不足,收敛速度变慢,影响了粒子群算法寻找最优解的时间. 同时避免算法在一些区域中盲目搜索, 为了更好的解决算法中在粒子最优速度和位置上的选择, 避免浪费时间, 引入惩罚函数

$$F(x,M)=M\bullet\sum_{i=1}^{m}[\min(0,g_i(x))]^2 \quad (10)$$

在公式(5)中，M 是一个惩罚因子，$\sum_{i=1}^{m}[\min(0,g_i(x))]^2$ 是一个惩罚项，当 $F(x,M)$ 的解逐渐靠近最优解的时候，$\sum_{i=1}^{m}[\min(0,g_i(x))]^2$ 必须满足当可行解满足的约束条件 $g_i(x)\ge 0$ 的时候，它的值非常小，反之，就需要进行使用惩罚函数，这样就能使得粒子个体的对速度和位置的优劣进行有效选择,从而来保证在搜索的过程中,避免无效区域的探索，从而产生最优解,更新公式如下:

$$\begin{aligned}v_{id}(t+1)&=w\times v_{id}(t)+c_1\times rand()\times(p_{best}-\\&\quad x_{id}(t))+c_2\times rand()\times(p_{gbest}-\\&\quad x_{id}(t))\ \ F(P_{best},g(x_{id}(t)))>\\&\quad F(P_{gbest},g(x_{id}(t)))\\x_{id}(t+1)&=x_{id}(t)+v_{id}(t+1)\ \ F(x_{best},g(x_{id}(t)))\\&\quad >F(x_{gbest},g(x_{id}(t)))\end{aligned} \quad (11)$$

当周围的粒子个体的速度和位置优于当前粒子的速度和位置的时候取 $w>1$,反之取 $w<1$. 其中 w 的取值如下:

$$w=\begin{cases}((iter_{\max}-iter_i)/iter_{aver})^{\alpha}\\((iter_i-iter_{\min})/iter_{aver})^{\beta}\end{cases} \quad (12)$$

代入公式(8)中，得到如下:

$$y_{ij}=x_{best,j}+w\bullet(x_{r1}-x_{r2}) \quad (13)$$

在公式(12)中,$iter_{\max}$ 表示最大迭代次数,$iter_{\min}$ 表示最小迭代次数,$iter_i$ 表示第 i 次搜索周围邻近的次数,α 表示一个常数,取值在[0,0.5]之间相对合理,β 表示一个常数,在[1,1.5]之间相对合理.

3.4 适应度函数

根据适应度函数，粒子可以选择下一代来进行搜索最佳的解决位置，适应度函数的合理的选取到算法的收敛速度以及解决方案的好坏,

在云计算任务调度中，总任务的完成时间和完成的成本是需要考虑的两个重要因素，因此，在云计算下的粒子群算法的总任务完成时间适应度函数和总成本适应度函数分别为:

$$f_t(i)=\frac{1}{sumTime_i} \quad (14)$$

$$f_c(i)=\frac{1}{sumCost(i)} \quad (15)$$

3.5 资源分配算法步骤

步骤1: 定义云计算中的任务调度目标函数，设定云计算任务相关参数.

步骤2: 设置初始化参数,种群数,最大迭代次数,最小迭代次数,其中设定粒子群的个体，种群的值的大小.

步骤 3: 计算粒子群的速度和位置,分别按照公式(6)和(7).

步骤 4: 当获得全局最优之后，执行公式公式(11)和(13)来进行局部最优的计算

步骤 5: 通过适应度函数(14)和(15)大小来计算当前粒子速度和位置与相邻的粒子的值的优劣，存储优于当前最优的粒子的速度和位置.

步骤 6: 循环次数加 1.

步骤 7: 满足终止条件,达到设定的循环次数.

步骤 8: 将获得最后存储位置的解就是云计算中最佳资源分配的结果，输出云计算中的资源分配最优解.

4 仿真实验

本实验采用运行环境为处理器为酷睿 i3，内存为 4GDDR3, Window7 操作系统，仿真软件为 Matlab2012, 云计算仿真平台为 Cloudsim. PSO 算法中设置种群的规模大小为 500，和的值为 1.4995，为 0.813，为 0.5，遗传算法设置种群设置为 1000，交叉概率为 0.9，变异概率为 0.02.

4.1 与基本 PSO 算法比较

设定虚拟任务为 100 个，虚拟资源为 10 个，通过比较两种算法下的时间和成本上的消耗可以发现，本文的算法时间花费低于基本 PSO 算法，主要是因为在 PSO 算法中引入了惩罚函数，使得算法的不必要消耗更多的时间在寻找无效的空间上，消除了算法初期震荡大的情况，同时通过遗传算法的引入，使得算法减少了产生最优解的时间，进一步解节约了成本，如图 1 和图 2 所示.

4.2 与其他智能算法进行比较

本文选取了文献[4]，文献[5]，文献[8]的算法与本

文的算法在收敛性以及云计算的条件下的任务–时间相比.

(1)算法收敛性比较

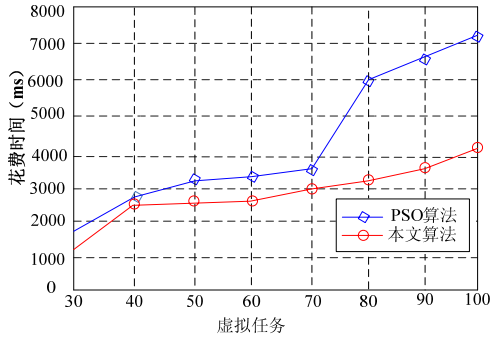

图 1　2 种算法时间消耗比较

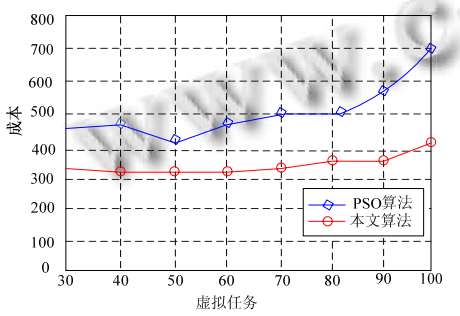

图 2　2 种算法成本消耗比较

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
$f_4(x)=-(\sum_{i=1}^{n}\lvert x_i\rvert)\exp(-\sum_{i=1}^{n}x_i^2)$	[–6,6]	30

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.

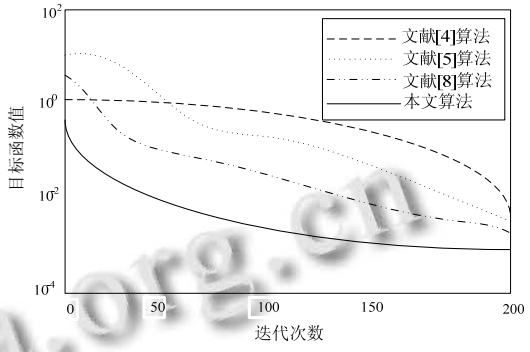

图 3　四种算法在 f_1 函数测试结果

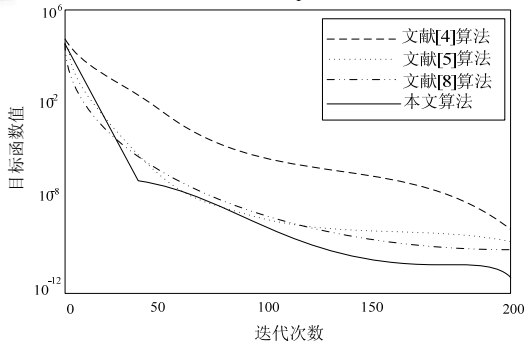

图 4　四种算法在 f_2 函数测试结果

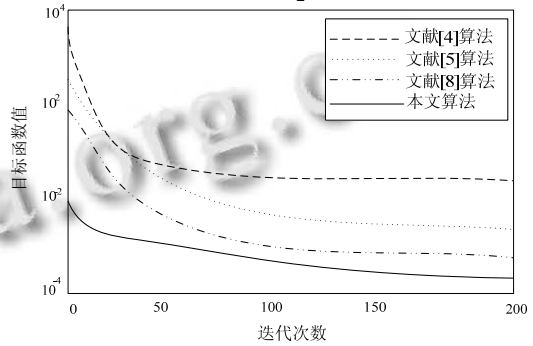

图 5　四种算法在 f_3 函数测试结果

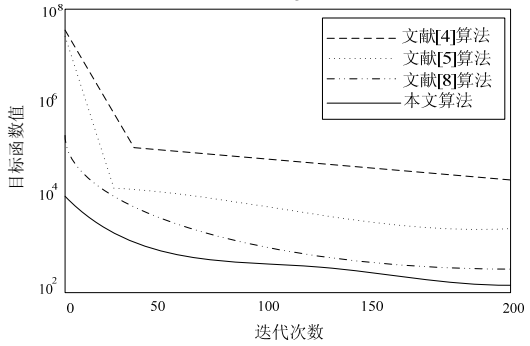

图 6　四种算法在 f_4 函数测试结果

(2)云计算下的任务-时间比较

设定虚拟任务为 800 个，虚拟资源为 10 个，设置迭代次数为 100. 比较结果如下

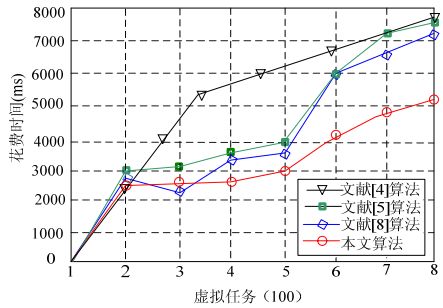

图 7 4 种算法的任务完成时间比较

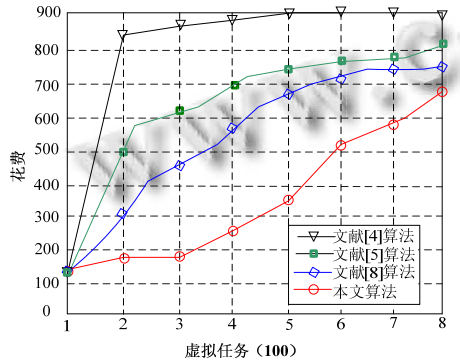

图 8 4 种算法的成本消耗比较

从图 7 中发现，本文算法在完成时间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 8 中发现经过改进后的本文算法在云计算环境中的成本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 结束语

在云计算资源算法分配的中引入基本粒子群算法，针对粒子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. 在粒子群种群寻找最优解中加入差分遗传算法，使得寻找过程中既能够发挥粒子群全局搜索快的优点，同时也能够发挥差分遗传算法局部搜索效率高的两者优点，同时引入惩罚函数避免了粒子向无效的空间移动. 仿真实验表明本文算法在一定程度上能够满足云计算资源分配要求，并且在任务完成时间，成本消耗等方面具有一定的优越性，为云计算的资源分配提供了一种参考.

参考文献

1 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

2 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

3 刘亚秋.云计算环境下基于时间期限和预算的调度算法.计算机工程,2013,39(6):56–59

4 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

5 郭力争.基于云计算环境下基于粒子群算法的多目标优化.计算机工程与设计,2013.34(7):2358–2362

6 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

7 朱宗斌,杜中军.基于改进的 GA 的云计算任务调度算法.计算机工程与应用,2013,49(5):77–80

8 丁燕艳等.云计算环境下的 PSO 可信资源调度.计算机工程与应用,2013,49(18):78–81