基于改进的蝙蝠算法在云计算中的资源分配①

宋芳琴

(绍兴职业技术学院 信息工程学院, 绍兴 312000)

摘 要: 云计算中的资源分配一直都是研究的重点, 提出了一种基于改进的蝙蝠算法的云计算资源分配方法. 在蝙蝠算法中引入差分遗传算法, 通过变异, 交叉和选择等操作避免个体陷入局部最优, 以及过早产生最优解的可能,改进后的蝙蝠算法能够有效的提高收敛速度和精度. 仿真实验表明, 本文算法不但有效提高了算法性能, 还优化了云计算系统中的资源调度能力, 提高了云计算资源的利用率.

关键词: 云计算; 资源分配; 蝙蝠算法; 差分遗传算法

Resources Allocation in Cloud Computing Based on Improved Bat Algorithm

SONG Fang-Qin

(Shaoxing Vocational & Technical College, Shaoxing 312000, China)

Abstract: Resource allocation in cloud computing has always been the focus of research, and in this paper, a resource allocation in cloud computing based on improved bat algorithm has been proposed. Differential genetic algorithm is introduced into bat algorithm and mutation, crossover and selection, etc. are employed to avoid individuals from falling into local optimum, and premature of the optimal solution. The improved bat algorithm can effectively improve the convergence speed and precision of the algorithm. Simulation experiments have shown that algorithm in this paper can not only greatly improve performance of the algorithm, but also optimize the resource scheduling capability in cloud computing system and improve utilization rate of resources in could computing.

Key words: cloud computing; resource allocation; bat algorithm; differential genetic algorithm

云计算是目前互联网络中最热门的探索方向, 它是分布式计算, 并行计算, 网格计算的发展产物[1]. 如何能够更加合理的分配云计算的资源成为了目前资源调度中的研究方向, 目前国内外学者进行了很多的研究. 文献[2]从虚拟机的角度出发, 建立一种云计算下的资源调度多目标综合评价模型,合理的将虚拟机的资源和任务合并为一个过程,降低复杂性. 文献[3]也是从虚拟机资源的角度出发, 根据不同的情况分别给出资源调度的策略,并验证算法的有效性. 文献[4]描述了分布式系统的资源分配, 在一定的程度上可以有效的提高云计算下服务质量. 近年来, 伴随着计算机智能技术的发展, 各种新的仿生算法逐一被提出, 比如遗传算法, 粒子群算法, 蜂群算法, 人工免疫算法,人工蛙跳算法[5-9]等,文献[10]提出在基于蛙跳算法的云计算中引入搜索策略, 避免算法早熟收敛, 有效的提高了云计算中的资源调度的效率. 文献[11]提出了一种具有双适应度的遗传算法(DFGA),实验结果表明,此算法是一种云计算环境下有效的任务调度算法. 文献[12]提出一种融合遗传算法与蚁群算法的混合调度算法, 该算法不仅克服了蚁群算法初期信息素缺乏,求解速度慢的问题,而且充分利用遗传算法的快速随机全局搜索能力, 使得蚁群算法在云计算资源模调度方面的优势得到了提高.

蝙蝠算法是近年来一种新型的智能算法, 在进行资源调度方面具有良好的效果, 文献[13-14]提出对蝙蝠算法进行改进, 一方面通过增加步长的方式来进行改进, 另一方面提出了一种混沌搜索策略的算法, 改进的算法为云计算的资源调度提供参考. 本文在蝙

① 收稿时间:2014-11-26;收到修改稿时间:2015-01-12

蝠算法的基础上, 将云计算的资源调度通过蝙蝠算法的优化来进行求解, 首先针对蝙蝠算法中缺乏一定的变异机制, 容易陷入局部最优的情况, 引入差分遗传算法使得改进后的蝙蝠算法全局寻优能力和搜索能力都得到增强, 通过测试函数的实验发现本文的算法的收敛速度和寻优精度上都有较大的提高. 仿真实验表明, 本文算法在云计算的调度方面提高了算法处理任务的效率, 减少了网络消耗时间.

1 云计算资源模型

如何能够进行云计算中的资源调度是解决云计算的关键问题, 云计算中的资源调度需要从虚拟节点的完成时间, 虚拟节点花费的网络费用, 虚拟节点的消耗网络资源等几个方面来进行考虑, 能够在多重解的情况的找到最小值, 本文为了简化研究问题的方便, 将以上的要求表述如公式(1)

$$\min_{t} F$$
$$F = s.t. \sum_{\substack{1\le i\le n \\ 1\le j\le m}} x_{ij}t_{ij} + x_{ij}c_{ij} + x_{ij}s_{ij} \tag{1}$$

其中, i 表示虚拟节点的数目且 $i \le n$ (n 为最大虚拟节点数目), j 表示资源的数目且 $j \le m$ (m 为资源总数目) x_{ij} 表示虚拟节点 i 在使用第 j 个资源, $\min F$ 表示在云计算环境下的完成完成资源分配时候的最小函数值, t_{ij} 表示 x_{ij} 的使用时间, c_{ij} 表示 x_{ij} 的使用网络费用, s_{ij} 表示 x_{ij} 的占据网络带宽资源.

2 蝙蝠算法

蝙蝠算法(Bat Algorithm, 简称 BA)[15]是 2010 年提出的一种新兴的启发式的智能算法. 该算法主要研究自然界中的蝙蝠的利用声纳来进行探测猎物, 从而能够避免障碍物的一种算法, BA 算法的仿生原理是将种群数量为 NP 的蝙蝠个体映射到 D 维空间中的 NP 的可行解, 将求可行解的适应度函数值的优劣来衡量蝙蝠算法中蝙蝠个体所处于位置的优劣, 将个体的优劣的过程模拟成算法优化和搜索过程中的使用优良的可行解来替代目前的可行解的过程. 在蝙蝠算法中, 具有以下两个规则:

为了能够花费最小的时间和空间代价找到食物的最优个体, 设定目标函数为 $\min f(x)$,目标变量根据前述的设置为 D 维空间, 因此目标变量为 $X = (x_1, x_2 \ldots x_D)^T$ 的优化问题, 因此 BA 算法实施过程如下:

设定蝙蝠的频率为 f_i, 其设定的范围是 $[f_{\min}, f_{\max}]$, 对应波长的范围是 $[\lambda_{\min}, \lambda_{\max}]$,蝙蝠响度为 A_0, 脉冲频率为 r ,第 i 只蝙蝠在 t 时刻的位置 l_i^t 和速度 v_i^t 的更新公式如下

$$f_i = f_{\min} + (f_{\max} - f_{\min})\alpha \tag{2}$$
$$v_i^t = v_i^{t-1} + (x_i^t - x^*) f_i \tag{3}$$
$$l_i^t = l_i^{t-1} + v_i^t \tag{4}$$

其中, f_i 表示的是蝙蝠频率, α 表示随机服从均匀分布的随机变量. x^* 表示前 i-1 次迭代之后获得的最优位置. 当进行局部搜索的时候, 从当前局部搜索中产生一个最优解, 每只蝙蝠就会随机产生一个新的位置.

$$l_{new} = l_{old} + \varepsilon \overline{A^t} \tag{5}$$

其中, ε 设定为随机参数, 为了避免蝙蝠位置的随意性扩大, 其取值在[-1,1]之间, $\overline{A^t}$ 表示在 t 时刻的所有蝙蝠的平均响度, 伴随着速度和位置的迭代而进行更新. 当食物发现之后, 响度下降, 脉冲频率上升.

$$A_i^{t+1} = \varepsilon A_i^t \tag{6}$$
$$r_i^{t+1} = r_i^0[1 - \exp(-\kappa t)] \tag{7}$$

由于蝙蝠算法能够在资源调度的效率方面优于其他的智能算法, 因此, 可以将蝙蝠算法运用到云计算的资源调度中, 在一定程度上解决云计算中资源分配不均, 同时提高资源分配效率.

3 基于改进的蝙蝠算法在云计算中的描述

3.1 蝙蝠算法在云计算中资源分配的不足

从以上蝙蝠算法中可以发现蝙蝠通过个体之间的作用和影响来确定当前寻找到最优食物的最优个体, 但在每次选择食物的过程中容易导致陷入局部最优以及收敛速度慢等情况, 特别是个体之间缺乏灵活机制, 出现的局部极值约束后无法跳出. 由于云计算中的虚拟节点在进行分配任务的时候存在很多的不确定的因素, 因此如何能够合理的分配到虚拟节点的资源就如同蝙蝠算法能够获得目标函数最小值的一样, 因此针对蝙蝠算法的改进在一定程度上可以模拟为针对云计算资源分配算法的改进. 由于蝙蝠算法在迭代的过程中, 种群个体迅速向周围进行靠近, 从而在一定程度上对于种群的规模进行减少, 同时导致多样性降低, 无

法获得最优解, 同时对云计算中的资源调度造成影响.

3.2 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具有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 $X_i=(x_{i1},x_{i2}\cdots x_{in})$ 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.

$$y_{ij}=x_{best,j}+F\bullet(x_{r1}-x_{r2}) \tag{8}$$

公式(8)中的 $x_{best,j}(t)$ 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 x_i 之间进行交叉, 得到新的个体

$$z_{i,j}=\begin{cases}y_{i,j}, & t\in[0,1]\\ x_{i,j}, & otherwise\end{cases} \tag{9}$$

公式(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 $z_{i,j}$ 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 $z_{i,j}$ 与上一代个体, $y_{i,j}$, 否则就保持 $y_{i,j}$ 不变, 直接进入下一代.

3.3 改进的蝙蝠算法在云计算中的描述

本文的算法改进如下: 对进化后的蝙蝠个体位置不是直接进入到下一次迭代中, 而是通过差分遗传算法中变异, 交叉和选择操作, 找到新的位置之后进行迭代. 本文考虑到初始位置时候的蝙蝠算法群体初期存在局部差异大的情况, 因此蝙蝠个体的位置采用公式(2)(3)(4)在进行获得, 同时对获得产生最优解进行随机扰动通过公式(8)(9)(10)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. 对蝙蝠算法中的各个参数进行初始化, 将云计算中的任务按照子任务进行划分对应生成蝙蝠种群, 并且子任务按照完成时间, 网络费用, 网络消耗带宽资源进行编码设定, 同时将蝙蝠个体设置为虚拟节点.

具体步骤如下:

步骤 1: 根据目标函数公式(1)来计算各个蝙蝠群个体的适应度的函数值, 从而确定当前所在位置的最优值对应的最优解(个体).

步骤 2: 通过公式(2)(3)(4)对蝙蝠算法中的频率, 速度和位置进行更新操作.

步骤3: 对当前产生的最优解进行随机扰动, 从而产生一个新解, 对公式(6)和(7)的 r 和 A 进行更新.

步骤 4: 对当前的最优解个体通过差分变异算法可以基于每一个蝙蝠个体的初始位置进行变异, 交叉和选择等操作, 从而可以得到新的蝙蝠位置.

步骤 5: 通过新的蝙蝠位置来获得个体适应度的新解, 通过与步骤 2 产生的新解进行比较, 从两者中获得更好的适应度的解

步骤 6: 完成规定的迭代次数之后, 算法停止. 因此对应的最优蝙蝠的一组个体的值就是最佳的一组虚拟节点, 在蝙蝠个体最优解就是这组虚拟节点上对应的云计算中资源调度的最优解.

4 仿真实验分析

4.1 本文算法性能测试

为了验证算法的有效性,采用文献[16]中的3个基准函数进行对比测试,从而验证算法的有效性以及算法的性能进行分析. 通过Windows xp的Matlab基础上进行测试.

1) Sphere 函数

$$f(x)=\sum_{i=1}^{n}x_i^2 \quad -100\le x_i\le 100$$

2) Schwefel 函数

$$f(x)=\sum_{i=1}^{N}(-x_i\sin(\sqrt{|x_i|})),-500\le x_i\le 500$$

3) Rosenbrock 函数

$$f(x)=\sum_{i=1}^{N-1}[100(x_i^2-x_{i+1})^2+(x_i-1)^2]\ -5\le x\le 5$$

本文从搜索值的质量效果出发,针对公式(8)的 α 和 ε 的值进行了设定,设定中3个函数中的 α 值和 ε 值都取 0.5. 三个函数的测试结果如表 1-3 所示,其中 aver 表示平均值,st 表示方差,min 表示最小值,max 表示最值,D 维取值为 3. 设定种群的规模为 50,70 和 100.

表 1 Sphere 函数测试结果

Sphere 函数		种群的数目为 50		种群数目为 70		种群数为 100	
		改进前	改进后	改进前	改进后	改进前	改进后
D=3	Aver	0.0005129	0.0004132	5.17E-10	6.95E-12	2.21E-16	1.19E-15
	St	0.0006312	0.0005172	3.19E-09	5.05E-12	3.74E-15	1.84E-17
	Min	0.0016145	0.0009137	4.13E-13	1.21E-13	1.78E-19	3.16E-18
	Max	0.0059761	0.0049761	6.81E-09	7.13E-12	2.27E-20	3.61E-18

表 2 Schwefel 函数测试结果

Schwefel 函数		种群的数目为 50		种群数目为 70		种群数为 100	
		改进前	改进后	改进前	改进后	改进前	改进后
D=3	Aver	2.4731459	4.752141	0.924698	0.685412	0.008025	0.006212
	St	2.8915210	3.045263	2.163254	1.871412	0.045621	0.023135
	Min	0.2418141	0.087452	0.004241	0.003254	0.291261	0.281276
	Max	9.3426341	9.212332	7.955712	7.864127	0.294578	0.192312

表 3 Rosenbook 函数测试结果

Rosenbrock 函数		种群的数目为 50		种群数目为 70		种群数为 100	
		改进前	改进后	改进前	改进后	改进前	改进后
D=3	Aver	3.725912	2.928742	1.524165	1.214589	1.15E-12	1.32E-13
	St	3.245217	2.901547	1.425417	1.325475	2.22E-13	0.84E-13
	Min	0.276985	0.238523	1.818745	1.412689	1.16E-15	1.57E-16
	Max	0.621457	0.694578	1.724521	1.627812	1.59E-15	1.51E-16

表 1-3 表明了本文随着种群数目的不同在收敛的精度和稳定性方面具有了一定的提高，由于云计算中的资源数目是非常庞大的，因此将种群数目模拟为相对较小的云计算的资源数目，可以为云计算资源调度算法提供一定的参考.

4.2 本文算法在云计算中任务分配

采用 CloudSim[17]平台进行测试,硬件主要包括酷睿 i3CPU 和 4GDDR3,Windows Xp, 软件采用 matlab2012 进行模拟. 设定虚拟任务为 100 到 300 个, 虚拟节点为 10 个, 虚拟资源数目为 1000, 设置迭代次数为 500. 将本文的算法和文献[2],文献[3]的算法在云计算模型中进行对比. 如图 1-5 所示

从图 1-2 中可以看出,本文算法对于资源的调度的优化效果要明显优于其他两种文献的算法. 从图 3-5 中任务数不同的时候的节点任务量的性能都非常稳定, 基本上节点都能获得资源 85%以上. 主要是因为改进后的蝙蝠算法能够有效的提高云计算模型中的资源的效率,伴随着任务数量增多,节点任务量的获得资源的稳定性逐步加强, 对于实际的云计算环境的资源具有一定的参考价值.

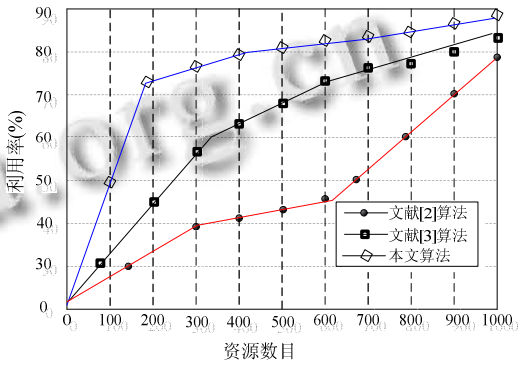

图 1 3 种资源负载算法资源利用率比较

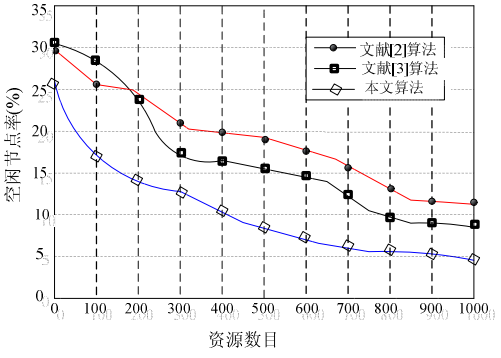

图 2 3 种资源负载算法能量消耗时间比较

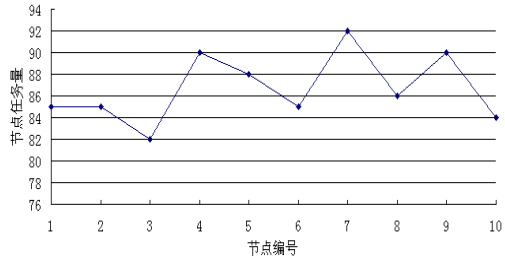

图 3 任务数为 100 的时候不同性能节点任务量

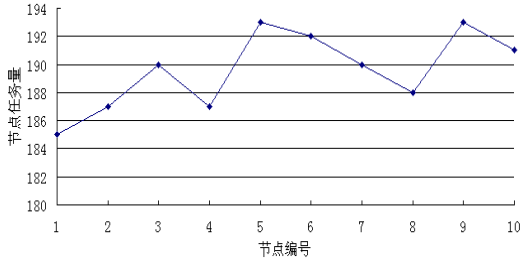

图 4 任务数为 200 的时候不同性能节点任务量

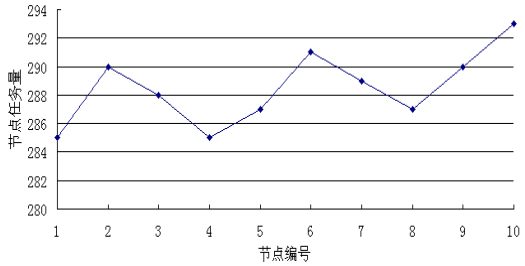

图 5 任务数为 300 的时候不同性能节点任务量

5 结论

云计算中如何能够合理使用资源是一个非常重要的问题. 本文在蝙蝠算法中,引入遗传差分算法, 使得改进后的算法具有全局和局部的搜索能力得到改善和提高, 提高了蝙蝠算法能够快速的寻找到优良的食物源的效率. 仿真实验表明, 本文算法有效的解决了资源分配的问题, 提高了算法分配资源的效率.

参考文献

1 Foster I, Zhao Y, Raicu I, et al. Cloud computing and grid computing 360-degree compared. Proc. of the 2008 Grid Computing Environments Workshop. Washington, DC. IEEEComputer Society. 2008. 1–10.

2 许波,赵超,祝衍军,等.云计算中虚拟机资源调度多目标优化.系统仿真学报,2014,26(3):592–595.

3 贲飞,汪芸.云计算下基于容错 QoS 的虚拟机资源分配策略.微电子学与计算,2013,30(3):136–139.

4 Abu-Rahmeh J, Talebbendiab A. A dynamic biased random sampling scheme for scalable and reliable grid networks. Jounal of Computer Science, 2008, 7(4): 1–10.

5 Goldberg D. Genetic Algorithms in Search. Optimization and Machine Learning, Reading. Mass:Addision-Wesley, 1989.

6 Kennedy J, Eberhart R. Particle swarm optimization. Proc. of IEEE International Conference on Neural Networks. 1995. 1942–1948

7 Dorigo M, Maniezzo V. Colorni A. The ant system: optimization by a colony of cooperating agents. IEEE Trans. on Systems, Man and Cybernetics-PartB, 1996, 26(1): 29–41.

8 Bersini H,Varela F. The immune recruitment mechanism: A selective evolutionary stategy. Proc. of the 4th International Conference on Genetic Algorithms. 1991. 520–526.

9 Teodorovic D. Bee colony optimization-a cooperative learning approach to complex transportation problems. Advanced or and AI Methods in Transportation. 10th EWGT Meeting Society Press. 1994, 11. 124–134.

10 赵鹏军,等.求解复杂函数优化问题的混合蛙跳算法.计算机应用研究,2009,26(7):2435–2437.

11 李建锋,彭舰.云计算环境下基于改进遗传算法的任务调度算法.计算机应用,2011,31(1):184–186.

12 周莲英.遗传算法与蚁群算法相融合的云计算任务调度算法研究.苏州大学,2013.

13 张宇楠,刘付永.一种改进的变步长自适应蝙蝠算法及其应用.广西民族大学学报(自然科学版),2013,19(2):51–54.

14 刘长平,叶春明.具有混沌搜索的蝙蝠算法及性能仿真.系统仿真学报,2013,25(6):1183–1188.

15 Yang XS. A new metaheuristicbat-inspored algorithm. In: Conzalez JR, et al. eds. Nature Inspired Cooperative Strategies for Ooptimization(NISCO2010). Berlin. Springer. 2010, 284. 65–74.

16 Karaboga D. An idea based on honey bee swarm for numerical optimization[Technical Report]. Erciyes University, Engineering Faculty, Computer Engineering Department, 2005. TR06.

17 Calheiros R, Ranjan R, Rose Ca Fd, Buyya R. CloudSim: a nove framework for modelingand simulation of cloud computing infrastructures and services[Technical Report]. Grid Computing and Distributed Systems Laboratory, The University of Melbourne, Australia, March 13, 2009. GRIDS-TR-2009-1.