Computer Science and Application 计算机科学与应用, 2018, 8(3), 286-295
Published Online March 2018 in Hans. http://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2018.83033

Hans汉斯

Research on Particle Swarm Optimization Algorithm for Solving Cloud Computing Task Scheduling

Qing Wang, Xueliang Fu, Gaifang Dong, Shasha Zhao

College of Computer and Information Engineering, Inner Mongolia Agricultural University, Hohhot Inner Mongolia
Email: 1206486506@qq.com

Received: Mar. 6th, 2018; accepted: Mar. 20th, 2018; published: Mar. 28th, 2018

Abstract

At present, task scheduling problem in cloud environment is a hot research topic, and particle swarm optimization algorithm (PSO) is an important intelligent algorithm to solve the task scheduling problem. According to the correlation, particle swarm Optimization algorithm (CPSO) and new adaptive inertia weight based particle swarm optimization algorithm (NewPSO) in solving this problem are easy to fall into the local optimal solution and poor searching ability, In this paper, the task execution time and cost as the goal, the random factor and the inertial weight are fused, and the Enhanced Particle Swarm Optimization (EPSO) is proposed. Simulation results show that under the same conditions, compared with PSO algorithm, CPSO algorithm and NewPSO algorithm, EPSO algorithm can reduce execution time and cost more effectively (including task execution time, time cost and virtual machine cost), and get a better scheduling solution.

Keywords

Task Scheduling, PSO Algorithm, Correlation, Execution Time, Cost Consumption

求解云计算任务调度的粒子群优化算法研究

王　晴，付学良，董改芳，赵莎莎

内蒙古农业大学计算机与信息工程学院，内蒙古　呼和浩特
Email: 1206486506@qq.com

收稿日期：2018年3月6日；录用日期：2018年3月20日；发布日期：2018年3月28日

文章引用: 王晴, 付学良, 董改芳, 赵莎莎. 求解云计算任务调度的粒子群优化算法研究[J]. 计算机科学与应用, 2018, 8(3): 286-295. DOI: 10.12677/csa.2018.83033

摘 要

目前，云环境下任务调度问题是一个研究热点，而粒子群算法(Particle Swarm Optimization, PSO)是解决任务调度问题的重要智能算法。针对相关性粒子群算法(Correlation Particle Swarm Optimization, CPSO)和新的基于自适应惯性权重粒子群算法(New Adaptive Inertia Weight Based Particle Swarm Optimization, NewPSO)在解决该问题时易陷入局部最优解和寻优能力差的不足，本文以任务执行时间和代价为目标，将随机因子与惯性权重相融合，提出增强型粒子群算法(Enhanced Particle Swarm Optimization, EPSO)。仿真结果表明，在相同条件下，与PSO算法、CPSO算法和NewPSO算法相比较，EPSO算法能更有效的减少执行时间，降低代价消耗(包括任务执行时间，时间花费以及虚拟机花费)，得到更优的调度方案。

关键词

任务调度，粒子群算法，相关性，执行时间，代价消耗

1. 引言

近年来，云计算以其独特的方式成为互联网发展的热门话题。云计算主要是利用虚拟化技术，将庞大的计算处理任务通过网络分成多个较小的子任务，再交给由多部服务器组成的庞大系统，经过计算后，将处理结果返回给用户[1] [2]。

在云计算中，任务调度策略直接影响到用户的任务执行效率以及云环境下资源的使用效率。在解决任务调度问题的研究领域里，群体智能优化算法越来越得到研究者的广泛关注，逐渐成为解决调度问题的新工具。1995 年，Kennedy 等提出 PSO 算法[3]。该算法的基本思想是粒子在寻优过程中，不仅考虑自身的局部认识，而且还要考虑种群的全局认识，即每个粒子是在充分利用这两个认知信息的基础上决定下一次的进化方向。由于 PSO 算法有参数少、易实现、收敛速度快等特性，已被广泛应用于各个领域。

文献[4]提出改进的粒子群算法优化神经网络及应用，通过采用动态惯性权重，提高粒子的多样性，一定程度上避免陷入局部最优。文献[5]提出一种基于种群多样性的粒子群优化算法设计及应用，通过种群多样性设计自适应惯性权重策略，实现粒子全局优化能力与局部开发能力的平衡，从而避免陷入局部最优解。文献[6]在传统惯性权重的基础上提出了新的基于自适应惯性权重的粒子群算法(New Adaptive Inertia Weight Based Particle Swarm Optimization，NewPSO)，可以进一步调整粒子速度以平衡粒子全局与局部搜索，使算法避免了陷入局部最优。文献[7]提出基于动态加速因子的粒子群优化算法研究，通过引入动态的加速因子，提高全局搜索能力，改善粒子群算法的收敛速度及精度。文献[8]提出优化粒子群的云计算任务调度算法，解决了粒子群算法在寻优过程中没有考虑随机因子作用而造成全局优化能力不足的缺陷。

上述方法只考虑惯性权重或者只考虑随机因子，并没有有效避免陷入局部最优解，提高全局优化能力。因此本文将自适应惯性权重与随机因子相融合，并将改进后的粒子群算法运用到云计算任务调度问题中，得到时间与代价均优的调度策略。

2. 云计算任务调度问题描述

随着云计算技术的发展，关于大规模任务调度的研究[9]越来越重要。而云计算任务调度主要就是研究如何为用户提交的任务分配资源，换句话说就是将多个相互独立的多样化任务分配到云中规模庞大的虚拟资源上，满足用户 QoS 要求、任务完成时间最小以及负载均衡最高等目标，其调度规模如图 1 所示。

3. 标准 PSO 算法

PSO 算法根据个体对搜索空间的适应度值大小对个体的优劣进行评价。假设有一种群，其粒子个数是 m，粒子的维数是 n，则有：第 i 个粒子的速度表示为 $v_i=\{v_{i1},v_{i2},\cdots,v_{id}\}$，$1\le i\le m$，$1\le d\le n$。第 i 个粒子的位置表示为 $x_i=\{x_{i1},x_{i2},\cdots,x_{id}\}$，$1\le i\le m$，$1\le d\le n$。第 i 个粒子的个体最优解表示为 $p_i=\{p_{i1},p_{i2},\cdots,p_{id}\}$，$1\le i\le m$，$1\le d\le n$。整个种群的全局最优解表示为 $p_g=\{p_{g1},p_{g2},\cdots,p_{gd}\}$。粒子速度和位置的更新方程如下：

$$v_{id}(t+1)=\omega v_{id}(t)+c_1r_1(t)\left(p_{id}(t)-x_{id}(t)\right)+c_2r_2(t)\left(p_{gd}(t)-x_{id}(t)\right) \tag{1}$$

$$x_{id}(t+1)=x_{id}(t)+v_{id}(t+1) \tag{2}$$

其中，ω 是惯性权重。c_1 和 c_2 是加速系数，通常设为 2 [10]。r_1 和 r_2 是[0, 1]区间内均匀分布的两个相互独立的随机数。要将 PSO 算法运用于云计算任务调度问题，需要对粒子进行编码。

3.1. 粒子编码、解码及其初始化

粒子编码方式有很多种，本文采用直接法。

假设有 k 个任务，t 个资源，且 $k>t$，当 $k=5$，$t=4$ 时，编码序列(4, 3, 3, 2, 1)即为一个粒子，直接法编码规则是将每 1 个任务对应 1 个资源，例如，任务 1 对应资源 4，任务 2 对应资源 3，任务 3 对应资源 3。

粒子解码目的是获取每个资源上运行的所有任务。例如，资源 3 上运行的所有任务有{2, 3}。定义矩阵 $time[i,j]$，用来记录任务 i 在资源 j 上的执行时间，则资源 j 上完成所有任务的总执行时间是：

$$Time(j)=\sum_{i=1}^{n}time[i,j]\quad(1\le j\le t) \tag{3}$$

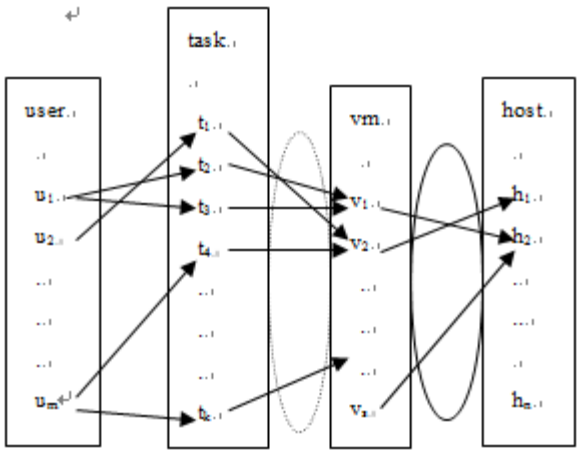

Figure 1. Cloud computing task scheduling model
图 1. 云计算任务调度模型

其中，i 表示资源 j 上执行的任务个数，n 表示资源 j 上执行的任务总数。则系统完成所有任务的总时间是：

$$TaskTime = \max\left(Time\left(j\right)\right)\ \ \left(1 \le j \le t\right) \tag{4}$$

然后进行种群初始化。假设有 k 个任务，t 个资源，s 个粒子，且 $k > t$。第 i 个粒子位置由向量 x_i 表示，定义为 $x_i = \left\{x_{i1}, x_{i2}, \cdots, x_{id}\right\}$。其中，$1 \le i \le s$，$1 \le d \le k$，$1 \le x_{ij} \le t$。第 i 个粒子速度由向 v_i 表示，定义为 $v_i = \left\{v_{i1}, v_{i2}, \cdots, v_{id}\right\}$。其中，$1 \le i \le s$，$1 \le d \le k$，$-r \le v_{ij} \le r$。初始化时产生 s 个粒子，每个粒子的位置和速度分别取 $\left[1, r\right]$ 和 $\left[-r, r\right]$ 间的随机整数。

本文主要研究如何减少执行时间和代价，因此适应度函数是：

$$fitness = 1/\left(TaskTime + time\cos t\right) \tag{5}$$

3.2. 目标函数

本文实验结果与其他算法比较的是执行时间和代价，因此将目标函数分别定义为：

$$TaskTime = FinishTime - StartTime \tag{6}$$

$$Cost = timecost + debt + TaskTime \tag{7}$$

其中，*TaskTime* 是执行时间，*FinishTime* 是任务调度结束时间，*StartTime* 任务提交时间。*Cost* 是代价，*timecost* 是时间花费(任务总长度/虚拟机的处理能力)，*debt* 是虚拟机的花费。

4. 相关性 PSO 算法(CPSO)

将随机因子 r_1 和 r_2 设为两个相互独立的随机数，没有考虑粒子在寻优过程中对个体最优解 p_i 和全局最优解 p_g 认知的联系。因此，需要建立 r_1 和 r_2 之间的相关性。

线性相关系数是一种常见的相关性分析[11]方法，其计算公式如下：

$$\rho = \frac{Cov\left(x, y\right)}{\sqrt{Var\left(x\right)Var\left(y\right)}} = \frac{E\left(\left(x - E\left(x\right)\right)\left(y - E\left(y\right)\right)\right)}{\sqrt{Var\left(x\right)Var\left(y\right)}}$$

由上式可知，要求随机变量 x 和 y 的相关系数ρ，必须知道期望与方差。但是，有些变量的期望与方差并不存在。而且，相关系数只能描述线性关系，不能描述非线性关系。Copula 理论为分析变量之间的相关性开辟了一条新思路。

所以，可通过二元 Copula 函数刻画变量 r_1 和 r_2 之间的相关性。

二元 Copula 的相关知识

定义 1 [12]：如果二元函数 $C:\left[0,1\right]^2 \to \left[0,1\right]$，满足下列两个条件：

(1)对于任意 $t \in \left[0,1\right]$，$C\left(t,0\right) = C\left(0,t\right) = 0$ 且 $C\left(t,1\right) = C\left(1,t\right) = t$；

(2)对于 $u_1, u_2, v_1, v_2 \in \left[0,1\right]$ 且 $u_1 \le u_2$，$v_1 \le v_2$，有 $C\left(u_2, v_2\right) - C\left(u_1, v_2\right) - C\left(u_2, v_1\right) - C\left(u_1, v_1\right) \ge 0$，则称函数 C 为一个 Copula。

定理 1 [12]：设随机变量 x，y 的联合分布函数为 $H:R^2 \to \left[0,1\right]$，其边缘分布函数分别为 F_x，F_y，则存在一个 Copula $C:\left[0,1\right]^2 \to \left[0,1\right]$，对任意 $x = \left(x_1, x_2\right) \in R^2$，有：

$$H\left(x_1, x_2\right) = C\left(F_x\left(x_1\right), F_y\left(x_2\right)\right) \tag{8}$$

由定理 1 可得以下推论。

推论 1 [12]：如果 $C:[0,1]^2 \to [0,1]$ 是一个 Copula 函数，F_x，F_y 分别是随机变量 x，y 的分布函数，那么存在一个以 F_x，F_y 为边缘分布的联合分布函数 H，满足对任意的 $(u_1,u_2)\in[0,1]^2$，有：

$$C(u_1,u_2)=H\left(F_x^{-1}(x_1),F_y^{-1}(x_2)\right) \tag{9}$$

定理 1 和推论 1 的作用是说明 Copula 函数的存在性及其生成方法。

Copula 函数类型很多，使用较多且应用广泛的一个是 Gaussian Copula，定义如下：

定义 2 [12]：对任意 $(u,v)\in[0,1]^2$，二元 Gaussian Copula 可定义为：

$$C_p(u,v)=\phi_p\left(\phi^{-1}(u),\phi^{-1}(v)\right) \tag{10}$$

其中，ϕ是标准正态分布函数，ϕ_p是二维正态变量的联合分布函数，ϕ^{-1}是ϕ的逆函数。

设随机变量 x 和 y 满足 $x,y\sim U(0,1)$，由推论 1 可知，它们的二元 Copula 函数实际上是 x 和 y 的联合分布函数。因此，用 Copula 函数研究随机因子 r_1 和 r_2 之间的相关性是可行的，具体定义如下：

$$H(r_1,r_2)=C(r_1,r_2) \tag{11}$$

其中，H 是 r_1 和 r_2 的联合分布函数，C 是相应的 Copula 函数。

定义 3：在 PSO 算法基础上，由式(1)、(2)和(11)共同描述粒子运动轨迹的算法称为 CPSO 算法。

CPSO 算法[8]具体实现过程如图 2 所示。其中，s 是种群规模，r 是虚拟机数量，t 是任务数量，ω是惯性权重，c_1 和 c_2 是学习因子，x 是迭代次数，ρ是相关系数。

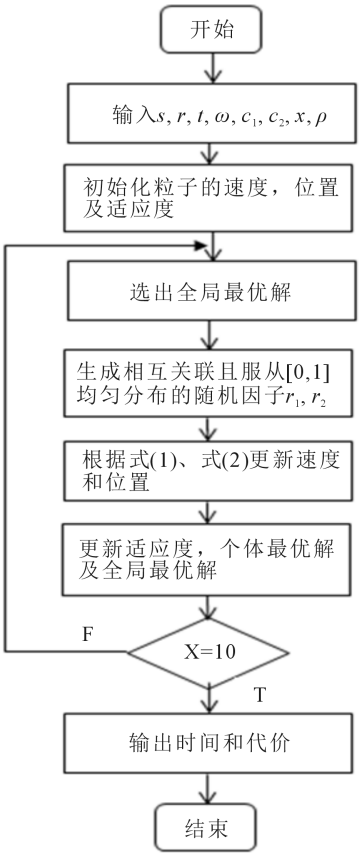

Figure 2. CPSO algorithm flow chart
图 2. CPSO 算法流程图

5. 新的自适应惯性权重粒子群算法(NewPSO)

通过每个粒子的适应度与全局最优值比较，可以更加准确地描述粒子所处的位置状态，成功值的计算方法如下所示：

$$S(i,t)=\begin{cases}1, & fitness\left(pbest_i^t\right)<fitness\left(pbest_i^{t-1}\right)\text{且}\ fitness\left(pbest_i^t\right)<globalbest^t\\ 0.5, & fitness\left(pbest_i^t\right)<fitness\left(pbest_i^{t-1}\right)\text{且}\ fitness(pbest_i^t)\geq globalbest^t\\ 0, & fitness\left(pbest_i^t\right)=fitness\left(pbest_i^{t-1}\right)\end{cases} \tag{12}$$

其中，$S(i,t)$是粒子 i 在第 t 次迭代过程中的成功值，$pbest_i^t$ 是粒子 i 在第 t 次迭代时的最优位置，$globalbest^t$ 是全局最优值，$fitness\left(pbest_i^t\right)$ 是粒子 i 在第 t 次迭代时最优位置的适应度。

$P_s(t)$是粒子群在第 t 次迭代时的成功率，表示本次迭代位置比上次好的粒子在粒子群中所占比重。其中，$\sum_{i=1}^{n}S(i,t)$ 表示所有粒子的成功值之和，n 表示整个粒子群的粒子个数。

$$P_s(t)=\sum_{i=1}^{n}S(i,t)\Big/n \tag{13}$$

$\omega(t)$是第 t 次迭代时的惯性权重，用于调整每次迭代时的粒子初速度。

$$\omega(t)=(\omega_{\max}-\omega_{\min})P_s(t)+\omega_{\min},\ 0\leq\omega_{\min}\leq\omega_{\max} \tag{14}$$

通过与全局最优值的比较更加准确地得出 $S(i,t)$。精确的 $S(i,t)$可以提高粒子群的成功率(式(13))和惯性权重(式(14))的自适应性，因此算法可以更准确地调整粒子速度，以平衡粒子的全局与局部搜索能力，避免陷入局部最优。

定义 4：在 PSO 算法基础上，且由式(1)、(2)和(12)、(13)、(14)共同描述粒子运动轨迹的算法称为 NewPSO 算法。

NewPSO 算法[6]具体实现过程如图 3 所示。其中，r_1 和 r_2 是随机因子，其它参数与上述 4.2 节中的参数含义相同。

6. 增强型粒子群算法(EPSO)

针对 CPSO 算法和 NewPSO 算法存在的缺点，本文提出在自适应惯性权重的基础上，适当融入随机因子之间的相关性，该算法称为 EPSO 算法。EPSO 算法不仅能提高粒子群的寻优能力，还可以避免陷入局部最优。EPSO 算法具体实现过程如图 4 所示。

7. 仿真实验与分析

7.1. 实验环境与参数

本文利用 CloudSim 构建云计算环境，Eclipse 作为开发平台，将本文提出的 EPSO 算法与基本 PSO 算法、CPSO 算法[8]以及 NewPSO 算法[6]进行对比分析。其中，算法参数见表 1。

7.2. 实验结果与分析

本文从任务执行时间和代价两个方面与其他三个算法进行比较。其中，执行时间和代价即 3.2 节中提到的。

7.2.1. 执行时间

实验设置任务数从 20 到 300 变化，单个任务的负载量是 30~3000 内的随机值[13]。分别取 20 到 100 个任务，迭代 100 次的时间变化如图 5 所示，横坐标代表任务数量，纵坐标代表时间。当任务数量在

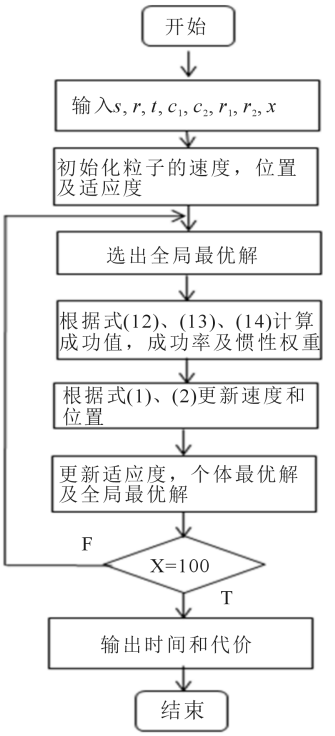

Figure 3. NewPSO algorithm flow chart
图 3. NewPSO 算法流程图

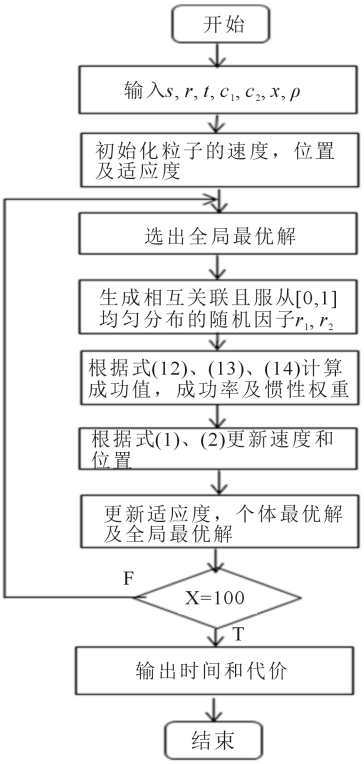

Figure 4. EPSO algorithm flow chart
图 4. EPSO 算法流程图

Table 1. EPSO algorithm parameters
表 1. EPSO 算法参数

参数名称	参数符号	参数取值
种群规模	s	100
资源数	r	5
任务数	t	50~300
权重系数	c_1, c_2	2
最大迭代次数	Iteration	100

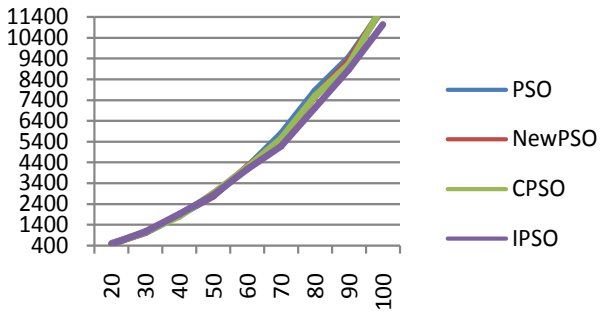

Figure 5. Change of time with tasks
图 5. 时间随任务的变化图

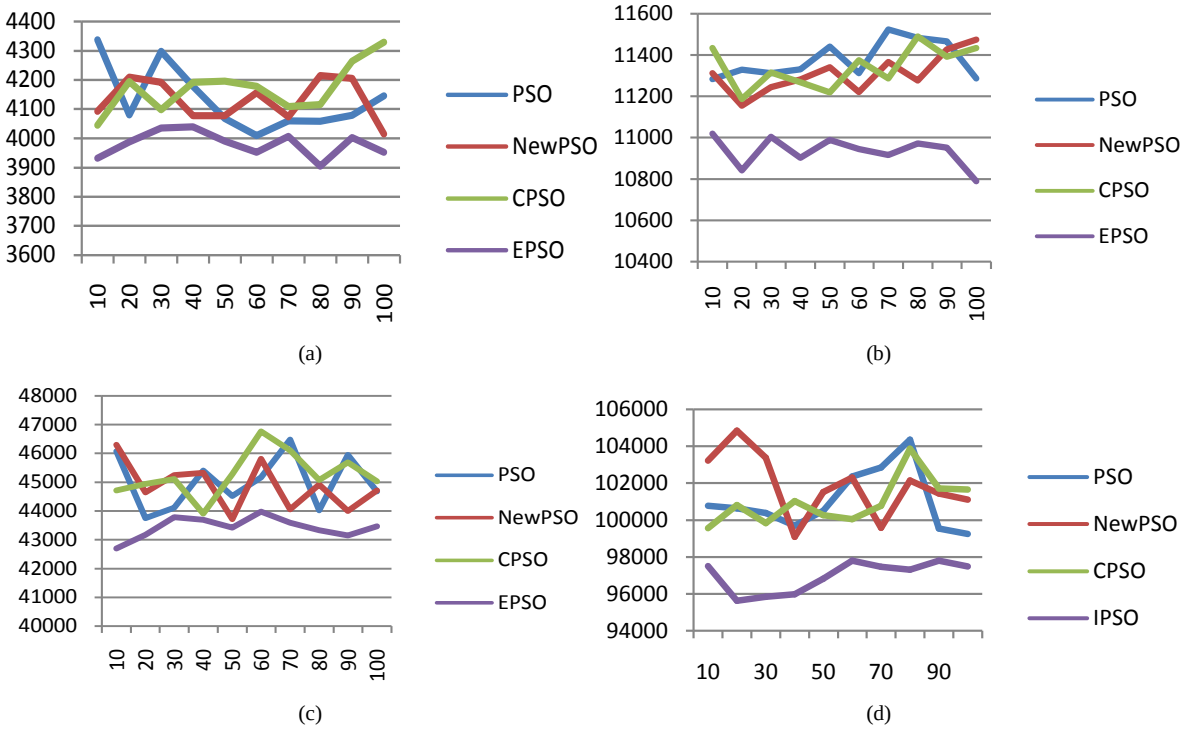

Figure 6. (a) 60-time change chart; (b) 100-time change chart; (c) 200-time change chart; (d) 300-time change chart
图 6. 任务数为 60 的时间变化图；(b) 任务数为 100 的时间变化图；(c) 任务数为 200 的时间变化图；(d) 任务数为 300 的时间变化图

20 到 50 时，四个算法并没有太大区别，当任务数量大于等于 60，迭代次数相同的情况下，随着任务数量的增加，EPSO 算法执行任务所花费的时间少于其他三个算法花费的时间，进一步说明 EPSO 算法适合海量任务的任务调度。图 6(a)~(d)分别是任务数为 60、100、200 和 300 时，迭代 100 次的时间趋势图，其中横坐标代表迭代次数，纵坐标代表时间。由图可看出，与其他三个算法相比，EPSO 算法的趋势较平缓，时间变化比较稳定。随着任务数量越多，EPSO 算法的优势越显著。总之，相比其他三个算法，任务越多，EPSO 算法越能在较短时间内找到最优解，提高全局优化能力。

7.2.2. 代价

分别取 20 到 100 个任务，迭代 100 次的代价，如图 7 所示，横坐标代表任务数量，纵坐标代表代价。当任务数大于等于 60 时，随着任务数量的增加，EPSO 算法的代价消耗低于其他三个算法。图 8(a)~(d)

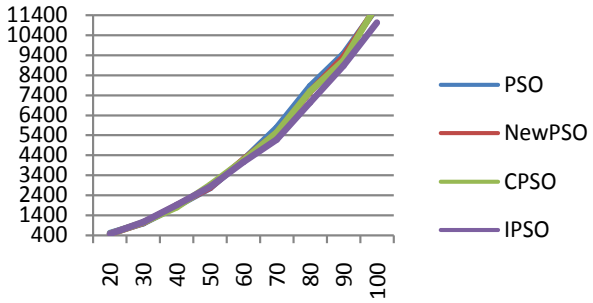

Figure 7. Change of cost with tasks
图 7. 代价随任务的变化图

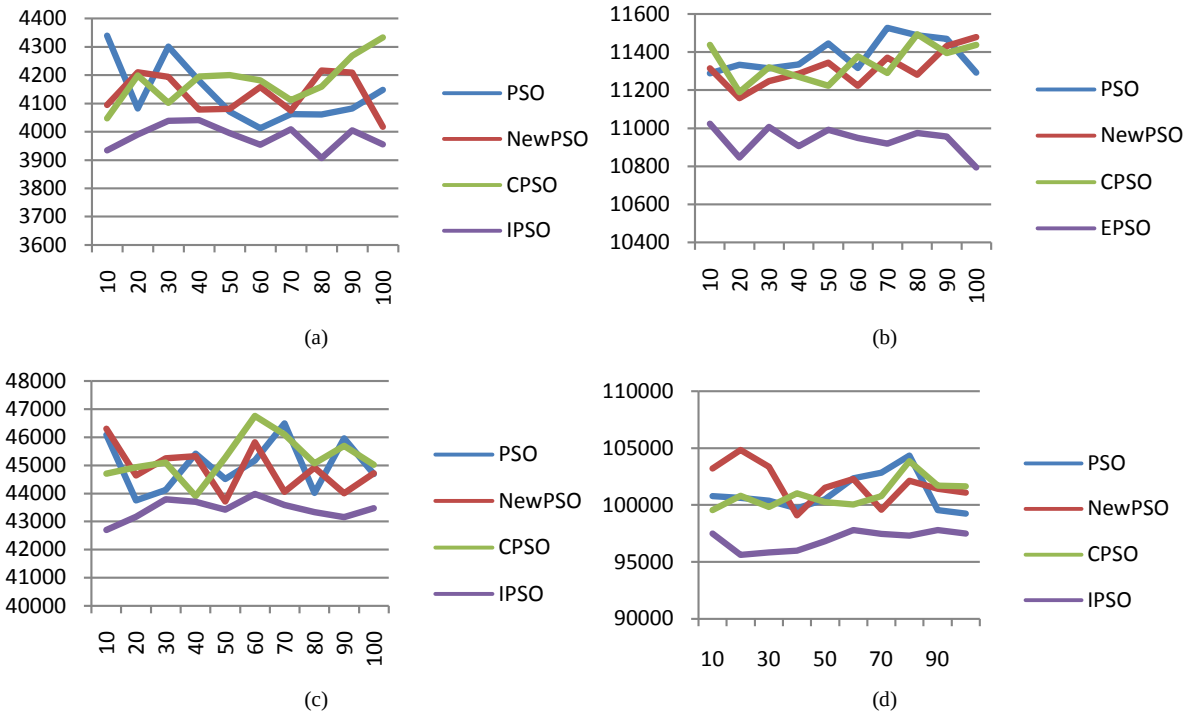

Figure 8. (a) 60-cost change chart; (b) 100-cost change chart; (c) 200-cost change chart; (d) 300-cost change chart
图 8. (a) 任务数为 60 的代价变化图；(b) 任务数为 100 的代价变化图；(c) 任务数为 200 的代价变化图；(d) 任务数为 300 的代价变化图

分别是任务数为 60、100、200 和 300 时，迭代 100 次的代价消耗趋势图，其中横坐标代表迭代次数，纵坐标代表代价。由图可知，随着迭代次数的增加，EPSO 算法的代价收敛趋于稳定，且低于其它算法。综上所述，与其他三个算法相比较，EPSO 算法在寻优过程中找到全局最优解的同时，所消耗的代价最少。

8. 结论

本文在自适应惯性权重的基础上，加入随机因子的相关性，提出 EPSO 算法。EPSO 算法可有效避免局部最优解，从而提高全局优化能力，获得时间与代价更优的调度方案。但是，本实验仅在模拟环境下实现，还需通过在实际的云计算平台上修改完善。

资助信息

国家自然科学基金(61063004，61363006)；内蒙古自然科学基金(2015MS0626，2015MS0605，2015MS0627)；内蒙古教育厅高校研究项目(NJZC059)；教育部留学人员基金([2014]1685)；内蒙古自治区科技计划项目(穿透降水量 GSM 网络在线监测与数据传输系统的研制)。

参考文献

[1] Abadi, D.J. (2010) Data Management in the Cloud: Limitations and Opportunities. *IEEE Data Engineering Bulletin*, **32**, 3-12.

[2] Lee, Y.C. and Zomaya, A.Y. (2012) Energy Efficient Utilization of Resources in Cloud Computing Systems. *Journal of Supercomputing*, **60**, 268-280. https://doi.org/10.1007/s11227-010-0421-3

[3] Kennedy, J., Eberhart, R.C. and Shi, Y. (2001) Swarm Intelligence. Elsevier Science Press, Singapore, 202-210.

[4] 何明慧, 徐怡, 王冉, 胡善忠. 改进的粒子群算法优化神经网络及应用[J/OB]. 计算机工程与应用, 2018, 1-9. http://kns.cnki.net/kcms/detail/11.2127.TP.20180205.1504.006.html

[5] 韩红桂, 卢薇, 乔俊飞. 一种基于种群多样性的粒子群优化算法设计及应用[J]. 信息与控制, 2017, 46(6): 677-684.

[6] 李学俊, 徐佳, 朱二周, 等. 任务调度算法中新的自适应惯性权重计算方法[J]. 计算机研究与发展, 2016, 53(9): 1990-1999.

[7] 滕志军, 吕金玲, 郭力文, 王志新, 许恒, 袁丽红. 基于动态加速因子的粒子群优化算法研究[J]. 微电子学与计算机, 2017, 34(12): 125-129.

[8] 谭文安, 查安民, 陈森博. 优化粒子群的云计算任务调度算法[J]. 计算机技术与发展, 2016, 7(26): 6-10.

[9] Arfeen, M.A., Pawlikowski, K. and Willig, A. (2011) A Framework for Resource Allocation Strategies in Cloud Computing Environment. *Proceedings of* 2011 *IEEE* 35*th Annual Computer Software and Applications Conference Workshops*, 18-22 July 2011, Washington, DC, 261-266.

[10] Netjinda, N., Sirinaovakul, B. and Achalakul, T. (2014) Cost Optional Scheduling in IaaS for Dependent Workload with Particle Swarm Optimization. *The Journal of Supercomputing*, **68**, 1579-1603. https://doi.org/10.1007/s11227-014-1126-9

[11] Embrechts, P., Mcneil, A.J. and Straumann, D. (2001) Correlation and Dependency in Risk Management: Properties and Pitfalls. *Proc. of the Risk Management*: *Value at Risk and Beyond*, Cambridge University Press, Cambridge, 176-223.

[12] Nelsen, R.B. (2006) An Introduction to Copulas. 2nd Edition, Springer-Verlag, New York, 10-28.

[13] Fard, H., Prodan, R. and Fahringer, T. (2013) A Truthful Dynamic Workflow Scheduling Mechanism for Commercial Multicloud Environments. *IEEE Trans on Parallel and Distributed Systems*, **24**, 1203-1212. https://doi.org/10.1109/TPDS.2012.257

知网检索的两种方式：

1. 打开知网页面 http://kns.cnki.net/kns/brief/result.aspx?dbPrefix=WWJD
   下拉列表框选择：[ISSN]，输入期刊 ISSN：2161-8801，即可查询
2. 打开知网首页 http://cnki.net/
   左侧“国际文献总库”进入，输入文章标题，即可查询

投稿请点击：http://www.hanspub.org/Submission.aspx
期刊邮箱：csa@hanspub.org