优化粒子群的云计算任务调度算法

谭文安[1,2],查安民[1],陈森博[1]
(1. 南京航空航天大学 计算机科学与技术学院,江苏 南京 210016;
2. 上海第二工业大学 计算机与信息学院,上海 201209)

摘 要: 任务调度作为云计算的关键技术之一,却一直没有得到很好的解决。针对云任务调度的特点,基于基本粒子群优化(PSO)算法,文中提出了一种带极值扰动的相关性粒子群优化(EDCPSO)算法。该算法采用 Copula 函数去刻画随机因子间的相关结构,支持粒子合理利用自身经验信息和群体共享信息,解决了粒子群优化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;采用添加极值扰动算子的策略,进一步改进粒子群优化算法,避免了粒子群优化算法在进化后期容易陷入局部寻优现象。仿真结果表明,在相同条件下,带极值扰动的相关性粒子群优化算法优于基本粒子群优化算法和 Cloudsim 原有调度算法,任务总的完成时间明显减少。

关键词: 任务调度;云计算;粒子群优化;相关性;极值扰动
中图分类号: TP301.6 **文献标识码:** A **文章编号:** 1673-629X(2016)07-0006-05
doi: 10.3969/j.issn.1673-629X.2016.07.002

Task Scheduling Algorithm of Cloud Computing Based on Particle Swarm Optimization

TAN Wen-an[1,2], ZHA An-min[1], CHEN Sen-bo[1]
(1. College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China;
2. School of Computer and Information, Shanghai Second Polytechnic University, Shanghai 201209, China)

Abstract: How to schedule cloud tasks efficiently is one of the important issues to be resolved in cloud computing. The Extremum Disturbed Correlative Particle Swarm Optimization (EDCPSO) algorithm based on basic Particle Swarm Optimization (PSO) is proposed for the characteristics of cloud environment. It uses the Copular function to measure correlation structures among random factors, support of particles properly using the individual experience and social sharing information, resolving the demerit that the PSO algorithm lacks of the global optimization ability because of not considering the function of the random factors in the optimization process. Moreover, it uses the strategy of adding extremum disturbed arithmetic operators to improve further the PSO, which resolves the demerit of falling into local extremum in the late evolution for PSO. Simulation shows that EDCPSO is better than PSO and Cloudsim original scheduling algorithm in the same experiment conditions. That is to say, the algorithm can reduce the total completion time of tasks.

Key words: task scheduling; cloud computing; PSO; correlation; disturbed extremum

1 概 述

为了解决“大用户”、“大数据”和“大系统”带来的一系列挑战,云计算技术应运而生。云计算作为当今信息领域的重大成果,发展迅速。任务调度是云计算的关键技术之一。一个好的任务调度算法,不仅有助于打造一个稳定、健壮、节能的云计算环境,还可以提高用户使用云计算服务的满意度。智能优化算法近年来受到广泛关注,成为解决调度问题的新工具。

1995 年,Kennedy 等提出粒子群优化(PSO)算法[1]。其基本思想是粒子在寻优过程中,不但要考虑自身的局部认识,而且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。由于该算法具有简单、高效、智能背景深刻等优点,被广泛应用于任务调度、机器学习、数据

收稿日期: 2015-10-15 **修回日期:** 2016-01-21 **网络出版时间:** 2016-06-22
基金项目: 国家自然科学基金资助项目(6127036);上海第二工业大学重点学科(XXKZD1301)
作者简介: 谭文安(1965-),男,博士,教授,CCF 高级会员,通讯作者,研究方向为软件构架技术、协同计算、服务计算与知识管理、信息化工程及其支持环境研究等;查安民(1987-),男,硕士研究生,研究方向为云计算、工作流调度与服务计算领域。
网络出版地址: http://www.cnki.net/kcms/detail/61.1450.TP.20160622.0842.004.html

聚类、流程规划等方面。

PSO 算法的数学描述如下:

设有一种群,其粒子的个数为 m ,粒子的维数为 n ,则有:

第 i 个粒子的速度表示为: $v_i = \{v_{i1}, v_{i2}, \cdots, v_{id}\}$, $1 \leqslant i \leqslant m, 1 \leqslant d \leqslant n$;

第 i 个粒子的位置表示为: $x_i = \{x_{i1}, x_{i2}, \cdots, x_{id}\}$, $1 \leqslant i \leqslant m, 1 \leqslant d \leqslant n$;

第 i 个粒子的个体最优解表示为: $p_i = \{p_{i1}, p_{i2}, \cdots, p_{id}\}$, $1 \leqslant i \leqslant m, 1 \leqslant d \leqslant n$;

整个种群的全局最优解表示为: $p_g = \{p_{g1}, p_{g2}, \cdots, p_{gd}\}$, $1 \leqslant i \leqslant m, 1 \leqslant d \leqslant n$ 。

粒子速度和位置的更新方程如下:

$$v_{id}(t+1) = \omega v_{id}(t) + c_1 r_1(t)(p_{id}(t) - x_{id}(t)) + c_2 r_2(t)(p_{gd}(t) - x_{id}(t)) \quad (1)$$

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

其中, ω 为惯性权重; c_1 和 c_2 为加速系数; r_1 和 r_2 为随机因子。

但是,要将 PSO 算法运用于云计算任务的调度,需要对其进行改进。

PSO 算法在搜索最优解过程中,需要利用个体最优解 p_i 和全局最优解 p_g 。由更新方程可知,二者的利用率依赖于加速系数和随机因子两大因素。为了提高 PSO 算法的全局寻优能力,文献[2-5]分别针对加速系数进行了改进。但是,目前很少有人对随机因子进行研究。在经典 PSO 算法中, r_1 和 r_2 是两个相互独立的随机数。为了深入研究粒子对 p_i 和 p_g 的利用,需要研究一下随机因子。

PSO 算法在迭代后期收敛速度变慢,这是 PSO 算法的一大缺点。文献[6]通过动态修改 ω 值,兼顾了搜索速度和精度;但对复杂的云计算任务的调度,这种通过改进惯性权重所达到的优化力度和实际效果都有待进一步提高。文献[7]在 PSO 算法中引入遗传算法,虽然在一定程度上达到了优化目的,但也增加了自身的复杂度。

文中通过优化 PSO 算法,提出了相关性 PSO(CP-SO)算法。该算法运用 Copular 函数建立随机因子之间的相关性,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。之后提出基于前者的带极值扰动的相关性 PSO(EDCP-SO)算法,解决了 PSO 算法在迭代后期收敛速度变慢的问题。

2　云计算任务调度问题描述

用户对调度结果的评价依据有很多,文中主要研究如何减少任务的完成时间。

2.1　粒子编码

目前存在多种编码方式,文中选择直接法。

设有 t 个任务, r 个资源,且 $t > r$ 。当 $t = 10$, $r = 3$ 时,编码序列(3,2,2,1,3,2,3,1,1,2)即为一个粒子。其中,每个任务对应一个资源,例如,任务 3 对应资源 1。

接着是对粒子进行解码,目的在于获取每个资源运行的所有任务。例如,资源 1 上运行的所有任务有{4,8,9}。

定义矩阵 matrix $[i,j]$,用于记录任务 i 在资源 j 上的执行时间。则资源 j 上完成所有任务的总时间为:

$$\text{resourceTime}(j) = \sum_{i=1}^{n} \text{matrix}[i,j] \quad (1 \leqslant j \leqslant r) \quad (3)$$

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。

系统完成所有任务的总时间为:

$$\text{taskTime} = \max(\text{resourceTime}(j)) \quad (1 \leqslant j \leqslant r) \quad (4)$$

2.2　种群初始化

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

2.3　适应度函数

文中主要研究如何优化能够减少任务的执行时间,因此将适应度函数定义为:

$$\text{fitness}(i) = \frac{1}{\text{taskTime}(i)} \quad (1 \leqslant i \leqslant s) \quad (5)$$

3　相关性 PSO 算法

3.1　随机因子的认知分析

根据 PSO 算法的心理学假设,随机因子体现了粒子在认知过程中的非理性行为,反映出粒子作为认知主体的情绪,而加速系数视为粒子对这种情绪的心理效应[8-9]。

由此可知: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,需要建立认知信息彼此之间的内在关系,即 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[9]方法,计算公式为:

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

从上式可知,要求随机变量 x 和 y 的相关系数,必须知道它们的期望与方差。然而,有些变量的期望和方差是不存在的。另外,相关系数不能用于描述非线性关系。Copula 理论为分析变量之间的相关性开辟了一条新的思路。该理论虽然早在 1959 年就诞生了,但是一直没有受到大家的关注。直到最近几年才被广泛应用于金融、保险、投资等领域的相关性分析。

3.2 基于 Copula 的相关性 PSO 算法

文中使用二元 Copula 函数可刻画两个变量 r_1 和 r_2 之间的相关性。当然,该函数可推广到多元形式。

3.2.1 二元 Copula 的相关知识

定义 1[10]:如果二元函数 $C:[0,1]^2 \to [0,1]$,满足下列两个条件:

(1) 对于 $\forall t \in [0,1]$,$C(t,0) = C(0,t) = 0$ 且 $C(t,1) = C(1,t) = t$;

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

Sklar 定理是一个非常重要的定理,下面给出它的数学描述。

定理 1[10]:设随机变量 x,y 的联合分布函数为 $H: R^2 \to [0,1]$,其边缘分布函数分别为 F_x,F_y,则存在一个 Copula $C:[0,1]^2 \to [0,1]$,对任意 $x = (x_1,x_2) \in R^2$,有:

$$H(x_1,x_2) = C(F_x(x_1),F_y(x_2)) \tag{6}$$

由定理 1 可得下面推论。

推论 1[10]: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:

$$C(u_1,u_2) = H(F_x^{-1}(u_1),F_y^{-1}(u_2)) \tag{7}$$

定理 1 和推论 1 的作用在于说明 Copula 函数的存在性以及它的生成方法。

Copula 函数类型很多,使用较多且应用较广的一个当属 Gaussian Copula,定义如下:

定义 2[10]:对任意 $(u,v) \in [0,1]^2$,二元 Gaussian Copula 可定义为:

$$C_\rho(u,v) = \Phi_\rho(\Phi^{-1}(u),\Phi^{-1}(v)) \tag{8}$$

其中,Φ 是标准正态分布函数;Φ_ρ 是二维正态变量的联合分布函数;Φ^{-1} 是 Φ 的逆函数;ρ 是随机变量间的线性相关系数,且 $1 \leqslant \rho \leqslant 1$。

3.2.2 CPSO 算法实现

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

$$H(r_1,r_2) = C(r_1,r_2) \tag{9}$$

其中,H 为随机因子 r_1,r_2 的联合分布函数;C 为相应的 Copula 函数。

定义 3:在 PSO 算法基础上,且由式(1)、(2)和(9)共同描述粒子运动轨迹的算法称为 CPSO 算法。

在 CPSO 算法中,两大随机因子是服从 [0,1] 均匀分布且相关的随机变量。因此,CPSO 算法实现的难点是如何生成满足给定 Copula 函数的 r_1 和 r_2。

CPSO 算法实现如下所述:

输入:随机产生的初始种群;

输出:全局最优解 p_g。

(1) 给出算法中所有参数的值;

(2) 按照 2.1 与 2.2 的要求对粒子进行编码并初始化种群;

(3) 根据已知的相关系数 ρ,按照步骤(4)~(7)生成相互关联且服从 [0,1] 均匀分布的随机数;

(4) 随机生成相互独立且服从 [0,1] 均匀分布的变量 u_1,u_2;

(5) 根据 $x = \Phi^{-1}(u_1)$,$y = \Phi^{-1}(u_2)$ 计算 x,y,其中,Φ 是标准正态分布函数;

(6) 根据 Cholesky 分解计算 x,y:$x = x$,$y = \rho x + (1-\rho^2)^{0.5} y$;

(7) 根据 $r_1 = \Phi(x)$,$r_2 = \Phi(y)$ 计算 r_1 和 r_2,则二者即为满足 Gaussian Copula 定义的两个相关随机数;

(8) 使用式(5)计算粒子的适应值;

(9) 更新粒子的个体最优解 p_i;

(10) 更新种群的全局最优解 p_g;

(11) 按照式(1)、式(2)分别更新粒子的速度和位置;

(12) 判断算法是否停止;

(13) 若否,则返回步骤(3);若是,取迭代停止时的 p_g 作为最优解。

4 CPSO 算法的多样性分析

4.1 粒子认知策略分析

由定义 2 及正态分布的对称性可以看出:

$$C_\rho(r_1,r_2) = \Phi_\rho(\Phi^{-1}(r_1),\Phi^{-1}(r_2)) = C_\rho(r_2,r_1) \tag{10}$$

上式表明,对于 $\forall \rho$,r_1,r_2 具有对称性,其对称轴为直线 $r_1 = r_2$。当 ρ 在 $[-1,1]$ 范围内逐渐增大时,r_1,

r_2 的取值逐渐向对称轴靠拢。当 $\rho = 1$ 时，r_1，r_2 的取值将会分布在对称轴之上。

CPSO 算法中，r_1，r_2 取值的对称性，反映出粒子在迭代过程中对 p_i 和 p_g 的利用率是相同的。随着 ρ 的增大，r_1，r_2 取值的集中分布性，反映出粒子在迭代过程中对 p_i 和 p_g 的利用率随 ρ 的增大而增大。

4.2　群体多样性分析

群体多样性反映了种群中每个粒子的差异性，是描述种群进化过程中粒子多样性的一个关键指标。

定理 2[11]: CPSO 算法中，种群多样性的期望与相关系数 ρ ($0 \leqslant \rho \leqslant 1$) 具有正比例关系。

由定理 2 可知，为了在寻优过程中始终保持种群较好的多样性，CPSO 算法应该给定较大的相关系数 ρ。

由定理 2 可得下面的推论:

推论 2: CPSO 算法中，种群多样性的期望正相关于粒子对个体最优解和全局最优解的利用率。

5　带极值扰动的 CPSO 算法

根据文献[11-13]可知，粒子在迭代过程中其位置期望收敛于 $\frac{c_1 p_i + c_2 p_g}{c_1 + c_2}$。在此基础上，如何进一步提高粒子位置的收敛极值成为人们需要考虑的一个问题。很多学者针对 PSO 算法后期的收敛性问题，提出了解决方案[14-15]。例如，在 PSO 中引入交叉算子、变异算子，提高 PSO 的自适应能力等。但是它们中的大多都有一个共同点，即通过改进 PSO 算法的不同参数来提高其收敛性。显然，这种改进力度和实际效果都有待进一步提高。

在 CPSO 算法中，个体最优解 p_i 和全局最优解 p_g 决定了粒子位置的收敛极值。如果每个粒子在逼近收敛极值的过程中不能发现比 p_g 更优的位置，则之后的所有迭代都将无效，因为进化过程已经结束。

5.1　极值扰动

文中选用种群迭代次数 t 作为触发条件，并且同时扰动个体最优解 p_i 和全局最优解 p_g，个体扰动算子定义为: $p_i = \alpha^{t_i > T_i} p_i$，全局扰动算子定义为: $p_g = \beta^{t_g > T_g} p_g$。其中，$t_i$ 和 t_g 分别表示 p_i 和 p_g 需要扰动的触发条件; T_i 和 T_g 分别表示 p_i 和 p_g 需要扰动的临界阈值。并且定义: $\alpha^{t_i > T_i} = \begin{cases} 1 & t_i \leqslant T_i \\ U(0,1) & t_i > T_i \end{cases}$，$\beta^{t_g > T_g} = \begin{cases} 1 & t_g \leqslant T_g \\ U(0,1) & t_g > T_g \end{cases}$。其中，$U(0,1)$ 为 [0,1] 均匀分布的随机数。

对式(1)和(2)添加扰动算子后变为:

$$v_{id}(t+1) = \omega v_{id}(t) + c_1 r_1(t)(\alpha^{t_i > T_i} p_{id}(t) - x_{id}(t)) + c_2 r_2(t)(\beta^{t_g > T_g} p_{gd}(t) - x_{id}(t)) \quad (11)$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1) \quad (12)$$

5.2　EDCPSO 算法实现

定义 4: 在 CPSO 算法基础上，由式(11)、(12)和式(9)共同描述粒子运动轨迹的算法称为带极值扰动的 CPSO(EDCPSO)算法。

修改 CPSO 算法的第 11 步: 按照式(12)、(13)对粒子的速度和位置进行更新，其他步骤不变，则可得到 EDCPSO 算法。

6　仿真实验与分析

6.1　仿真实验环境与算法参数

利用 Cloudsim 构建云计算环境，Eclipse 作为开发平台，将文中提出的 EDCPSO 算法与 Cloudsim 原有调度算法 FIFO、基本 PSO 算法进行对比分析。其中，算法参数见表 1。

表 1　EDCPSO 算法参数

参数名称	参数符号	参数取值
种群规模	s	150
资源数量	r	50
任务数量	t	50、100、500
惯性权重	w	0.9
学习因子	c_1	2
	c_2	2
最大迭代次数	$t_{\max}$	200
临界阈值	T_i	3
	T_g	5

6.2　实验结果和性能分析

每个实验分别设置 5 组不同的用户任务、50 个资源节点，实验过程中需要记录每次实验结束后任务的总完成时间。所有实验各运行 20 次，取 20 次的平均结果作为最终结果。最后得出各个算法的任务完成时间，如图 1 ~ 3 所示。由图可见，EDCPSO 算法相比其他两种调度算法更加高效。

7　结束语

文中在分析云计算环境及其 PSO 算法的基础上，提出了带极值扰动的相关性 PSO(EDCPSO)算法。仿真结果表明: 该算法相比于 Cloudsim 原有调度算法和基本 PSO 调度算法，具有更高的效率，能够减少任务总的完成时间。

文中工作处于理论分析与仿真测试阶段，还需在实际的云计算平台上应用验证; 同时EDCPSO算法只

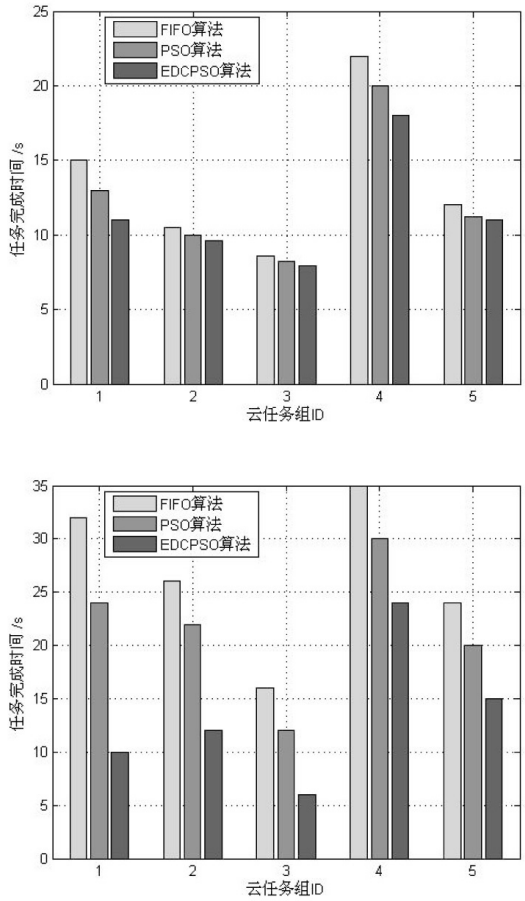

图 1 t =200时的任务总完成时间

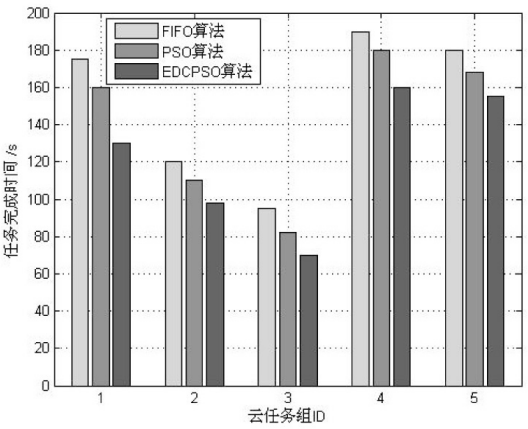

图 3 t =500 时的任务总完成时间

是优化了任务的完成时间，而在真实环境中还需考虑其他因素，如平均完成时间、经济效益、负载均衡等。因此，有必要进一步研究和改进该算法，以便拓宽其应用场景。

参考文献：

[1] Kennedy J, Eberhart R C, Shi Y. Swarm intelligence [M]. Singapore: Elsevier Science Press, 2001: 202-210.

[2] Parsopoulos K E, Vrahatis M N. Recent approaches to global optimization problems through particle swarm optimization [J]. Natural Computing, 2002, 1(2-3): 235-306.

[3] Ratnaweera A, Halgamuge S K, Watson H C. Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients [J]. IEEE Transactions on Evolutionary Computation, 2010, 8(3): 240-255.

[4] Arumugam M S, Rao M V C, Tan A W C. A novel and effective particle swarm optimization like algorithm with extrapolation technique [J]. Applied Soft Computing, 2009, 9(1): 308-320.

[5] 介 婧，曾建潮，韩崇昭. 基于群体多样性反馈控制的自组织微粒群算法 [J]. 计算机研究与发展，2008，45(3)：464-471.

[6] Shi Y, Eberhart R C. Fuzzy adaptive particle swarm optimization [C] //Proc of the congress on evolutionary computation. Piscataway: IEEE, 2001: 101-106.

[7] Angeline P J. Evolutionary optimization versus particle swarm optimization: philosophy and performance differences [C] // Proc of the 7th annual conference on evolutionary programming. Berlin: Springer-Verlag, 1998: 601-610.

[8] 王 沛. 社会认知心理学 [M]. 北京：中国社会科学出版社，2006：72-86.

[9] Embrechts P, Mcneil A J, Straumann D. Correlation and dependency in risk management: properties and pitfalls [C] // Proc of the risk management: value at risk and beyond. Cambridge: Cambridge University Press, 2001: 176-223.

[10] Nelsen R B. An introduction to copulas [M]. 2nd ed. New York: Springer-Verlag, 2006: 10-28.

[11] 申元霞，王国胤，曾传华. 相关性粒子群优化模型 [J]. 软件学报，2011，22(4)：695-708.

[12] Shi Y H, Eberhart R C. Population diversity of particle swarms [C] //Proc of the IEEE world congress on computational intelligence. Hong Kong: IEEE Press, 2008: 1063-1067.

[13] Clerc M, Kennedy J. The particle swarm: explosion stability and convergence in a multi-dimensional complex space [J]. IEEE Transactions on Evolution Computer, 2002, 6(1): 58-73.

[14] 吕振肃，侯志荣. 自适应变异的粒子群优化算法 [J]. 电子学报，2004，32(3)：416-420.

[15] 胡 旺，李志蜀. 一种更简化而高效的粒子群优化算法 [J]. 软件学报，2007，18(4)：861-868.