文章编号: 1001-0920 (2015) 12-2121-08 DOI: 10.13195/j.kzyjc.2014.1554

基于诺兰模型思想的改进混沌粒子群优化算法及评价

戴婉仪, 张 梅, 吴凯华, 胡跃明

(华南理工大学 a. 自动化科学与工程学院，b. 精密电子制造装备教育部工程研究中心，广州 510641)

摘 要: 针对粒子群优化算法 (PSO) 在处理高维复杂函数时容易陷入局部极值、收敛速度慢的缺陷, 从系统的认知分析过程和角度出发, 提出一种基于诺兰模型 (NM) 思想的改进 PSO 算法. 该算法在 Tent 混沌映射选择的参数的基础上, 结合 NM 信息融合和协调的思想, 在速度更新过程中增加均衡项, 并设计粒子群的欧氏距离指数以防止早熟, 从而实现对粒子的自动调整、保证多样性和提高算法的全局搜索能力. 最后, 运用典型函数对所提出算法进行测试, 并与最新相关算法进行比较, 结果表明, 所提出算法在全局搜索能力、效率和稳定性方面均具有明显的优势.

关键词: 粒子群优化；群智能；混沌；诺兰模型

中图分类号: TP18 **文献标志码:** A

Improved chaotic particle swarm optimization algorithm and valuation based on Nolan model thinking

DAI Wan-yi, *ZHANG Mei*, *WU Kai-hua*, *HU Yue-ming*

(a. College of Automatic Science and Engineering，b. Engineering Research Centre for Precision Electronic Manufacturing Equipments of Ministry of Education，South China University of Technology，Guangzhou 510641，China. Correspondent：ZHANG Mei，E-mail：zhangmei@scut.edu.cn)

Abstract: Aiming at the problem that the particle swarm optimization(PSO) algorithm trends to trap in local extreme, and performs high dimensional complex functions inefficiently, from perspectives of the cognitive analysis process of the system, a PSO algorithm based on Nolan model is proposed. The Tent chaotic map is introduced to improve the ergodicity of the algorithm, and the Euclidean distance index is given based on particle average position to automatically adjust particles' position and ensure diversity to improve the global search capacity of the algorithm. Finally, typical functions are used to test the proposed method. Compared with the current algorithms, it is showed that the proposed method has the advantages in global search, efficiency and stability.

Keywords: particle swarm optimization；swarm intelligence；chaotic optimization；Nolan model

0 引 言

粒子群优化算法 (PSO) 是一种基于种群的随机优化方法, 最初由 Eberhart 等[1-2]提出并成功应用于全局优化问题的寻优. 它与同为基于仿生学的遗传算法 (GA) 和蚁群智能优化算法 (ASO) 相比, 机理更为简单, 容易实现, 收敛速度较快. 因此, 近年来 PSO 算法得到了研究者们的广泛关注, 主要集中在实际工业应用上, 如任务分配[3]、电力系统[4]、通信[5-6]、自动控制[7-8]、经济[9]、生物医学[10-11]等, 同时, 在典型理论问题研究如离散型组合优化[12]、约束条件下处理机制优化[13]、多目标组合优化[14]、动态环境调节优化[15]等领域也有较大进展. 然而, 研究者在应用中发现粒子群算法极易陷入局部最优, 出现早熟现象. 为克服这些缺点, 研究者提出增加粒子群的规模以扩展解空间、双量子粒子群算法[16]、引入混沌思想[17-18]、将人类的创造性思维引入 PSO (CTPSO)[19], 这些改进使算法收敛速度和收敛精度得到了很大提升. 但是, 面对复杂程度越来越高的优化问题, 已有算法在寻优质量和优化速度上依然不尽人意.

本文从全新的系统认知过程和角度分析 PSO 算法, 将著名的信息系统进化模型——诺兰模型 (NM)[20] 的 6 个阶段引入到 PSO 算法 (NMPSO) 中. NM 强调

收稿日期: 2014-10-13；**修回日期:** 2015-01-27.

基金项目: 广东省产学研重点项目(2011A090200047)；广州市科技重大专项计划产学研专项项目(2012Y5-00004)；中央高校基本科研业务费专项项目(x2zdD2153910).

作者简介: 戴婉仪(1971−), 女, 副教授, 博士生, 从事控制理论与控制工程的研究；胡跃明(1960−), 男, 教授, 博士生导师, 从事智能优化算法等研究.

了信息资源整合的重要性, 进化过程中的信息充分融合不仅是生物体感知环境和行为行动的基础, 而且是生物体生存、进化和发展的基本能力要素[21]. 本文借鉴NM六阶段模型构建NMPSO算法框架, 为保证算法搜索的遍历性, 利用混沌映射确定PSO的惯性因子和加速因子. 基于NM中对信息集成、平衡和协调的思想, 利用粒子本身记忆信息、个体局部信息和群体间的共同信息增加信息均衡项, 改进PSO的速度和位置更新公式. 另外, 设计了粒子群欧氏距离指数, 由其判断并保证搜索初期粒子的多样性, 提高粒子的全局搜索能力. 为了验证所提出的NMPSO的寻优性能, 选取6个Benchmark标准函数进行寻优测试, 最后给出与最新相关算法的对比测试结果和分析.

1 PSO算法描述

PSO是一种群智能优化模型, 利用速度和位置更新公式, 通过个体间的协作与竞争寻找最优解. 在PSO求解优化过程中, 问题的解对应于搜索空间粒子的位置. Shi等[22]提出了带惯性参数的PSO速度更新公式, 其基本原理为: 假设搜索空间为n维, 总的粒子数为N, 粒子i表示为

$$x_i = (x_{i1}, x_{i2}, \cdots, x_{in})_{(i=1,2,\cdots,N)},$$

经过进化后所经历的最好的位置成为局部最优解, 记为$L_i = (L_{i1}, L_{i2}, \cdots, L_{in})$; 若$\exists g$, 使$L_g$是$L_i(i = 1, 2, \cdots, N)$中的最优, 则称其为全局最优解, 记为$G = (G_1, G_2, \cdots, G_n)$; 粒子$i$的速度表示为$v_i = (v_{i1}, v_{i2}, \cdots, v_{in})$. 在第$j$维上, 粒子$i$的速度和位置更新方程[21]分别为

$$v_{ij}^{k+1} = \lambda v_{ij}^k + c_1 r_1 (L_{ij}^k - x_{ij}^k) + c_2 r_2 (G_j^k - x_{ij}^k), \quad (1)$$

$$x_{ij}^{k+1} = x_{ij}^k + v_{ij}^k. \quad (2)$$

其中: $r_i = \mathrm{rand}(0,1)$, $i = 1, 2$; k为迭代次数; λ为惯性因子, 在迭代过程中, 动态调整λ取值可显著改善算法的收敛性能, 例如文献[23]提出随着k增大, 将λ从0.9线性递减至0.4, 文献[24]运用随机近似理论分析粒子群的动态行为, 提出随着k增大, 将λ递减至0; c_1和c_2为学习因子(加速因子), 用于调整粒子自身经验和群体经验在整个寻优过程中的权重, 一般取为常数; L_{ij}^k为第i个粒子第k次迭代时粒子局部最优的第j维值; G_j^k为整个粒子群在进行第k次迭代时全局最优解的第j维值.

2 基于诺兰的阶段模型的PSO算法

2.1 诺兰的阶段模型的引入

Nolan通过对信息系统的实践和经验的总结, 得到信息进化的6个阶段模型, 即诺兰模型[20]. 无论对于一个企业还是对于一个国家或地区而言, 信息化大致要经历初始、蔓延、控制、集成、数据管理和成熟等阶段, 且必须从一个阶段发展到下一个阶段, 不能实现跳跃式发展. 诺兰模型强调企业和机构信息资源整合的重要性, 同时应根据具体问题确定解决方案. 将诺兰模型思想引入PSO算法, 克服了信息资源整合不足制约信息交流和有效利用的问题, 实现种群内信息充分共享, 防止粒子因缺乏足够的信息陷入局部极值, 使粒子群能够“持续进化”, 抑制早熟停滞.

诺兰模型中集成阶段和数据管理阶段所体现的整合信息资源, 从单向应用转变为综合全面分析解决问题的思想, 与生物群体寻求一致的认知过程类似, 同时也与标准PSO速度更新公式(1)的本质相吻合. 式(1)中第1项为惯性项, 表示粒子对上一代速度的记忆, 代表粒子有维持自己先前速度的趋势; 第2项为自身学习项, 表示粒子吸取自身历史最佳经验并向其位置逼近的过程; 第3项为群体学习项, 表示粒子倾向群体历史最佳位置, 以达到粒子间合作和知识共享的目的. 个体在更新时不仅根据惯性项和自身学习项考虑自己的观点, 而且根据群体学习项向其他个体的观点进行适应性调整, 力求更好地综合、平衡和协调自身信息.

NMPSO算法的基本思路是: 每个粒子都具有个体本身的局部信息, 与群体间的相互作用得到均衡信息, 根据所得到的共同信息进行组合、评估和变换, 并在外部激励的影响下, 给出均衡项, 使得粒子群的整体搜索能力得到提升.

2.2 NMPSO算法框架和分析

NMPSO算法的每个过程与诺兰模型的初始阶段、传播阶段、控制阶段、集成阶段、数据管理阶段和成熟阶段6个阶段相对应, 算法框架如图1所示:

1) 在初始阶段, 每个粒子获得当前速度和位置, 并得到自身的最好位置和群体的最好位置;

2) 在传播阶段, 粒子搜索领域随着迭代次数的增加从自身逐步扩大到整个种群, 由于惯性因子λ可以有效地平衡PSO算法的局部和全局搜索能力, 对算法收敛性起着非常重要的作用, 同时考虑到混沌的敏感性和遍历性等优点, 将混沌映射引入到惯性因子中进行优化;

3) 在控制阶段, 粒子间进行信息交换, 每个粒子除了学习自身领域最佳历史位置外, 还学习邻域内其他粒子的成功经验, 在PSO算法中, 参数r_1、r_2、c_1、c_2也是影响算法收敛的重要因素, 因此在NMPSO算法中, 其取值全部采用混沌序列, 使r_1、r_2、c_1和c_2取得的值具有更好的遍历性, 加强学习因子c_1和c_2对每个粒子趋向最优位置运动的控制作用;

4) 在集成阶段, 结合混沌算子和粒子群算法, 重新设置粒子的速度和位置的表达式;

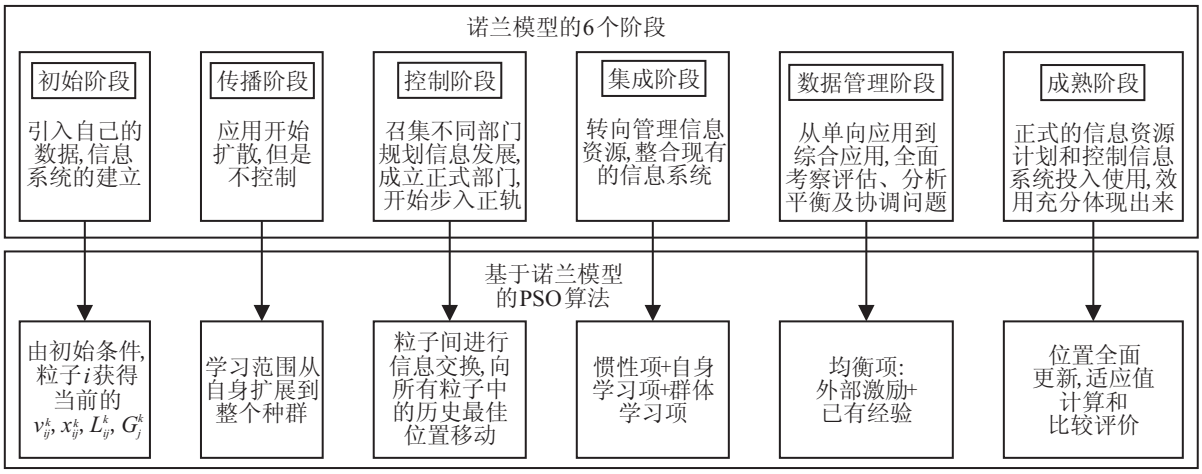

图 1　NMPSO 算法框图

5) 数据管理阶段, 需要全面考虑收敛速度和整体搜索能力, 根据以往的经验在外部激励作用下增加均衡项;

6) 在成熟阶段, 为寻求得到最优解, 不但要提升整体的搜索能力和收敛速度, 还要防止陷入局部极小值和“早熟”, 在搜索初期, 粒子群容易向当前最优位置聚集, 无法在搜索范围内重新搜索, 导致出现“早熟收敛”现象, 陷入局部极小值, 因此给定早熟判定条件, 进行早熟处理十分重要, 由于粒子的位置决定了粒子的适应度, 将粒子的位置的状态作为早熟收敛判断的条件.

2.3　NMPSO 算法设计

NMPSO 算法引入诺兰模型的思想, 对标准 PSO 进行增加早熟处理, 添加均衡项, 并运用混沌算子设置惯性因子和学习因子的改进.

2.3.1　早熟处理

NMPSO 算法利用粒子群平均位置给出欧氏距离指数, 从而进行早熟判断. 设 α^2 为粒子群的群体位置欧氏距离指数, 定义为

$$\alpha^2 = \sum_{i=1}^{m} \frac{D(x_i)}{f}. \tag{3}$$

式 (3) 通过参数 f 实现对距离指数 α^2 的归一化. $D(x_i) = \sqrt{\sum_{j=1}^{N}(x_{ij} - \overline{x}_j)^2}$ 为第 i 个粒子的欧氏距离, x_i 为当前的粒子的位置, $\overline{x}$ 为粒子群目前的平均位置, f 取值如下:

$$f = \begin{cases} \max\limits_{i=1}^{n} D(x_i), & \max\limits_{i=1}^{n} D(x_i) > 1; \\ 1, & \text{otherwise}. \end{cases} \tag{4}$$

式 (3) 中, 粒子群的欧氏距离指数 α^2 反映了粒子群离散程度, 其值越大, 表明离散程度越大, 反之, 表明粒子群较为集中. 若粒子过于集中, 则会降低种群多样性, 使算法陷入早熟. 在算法初期, 当 $\alpha^2 < c$ (c 为一个给定常数) 时, 需要提高粒子群体多样性, 对其进行早熟处理. 即 $\exists D(x_i) < fc/m$, 更新其位置为

$$x_{ij}^{k+1} = x_{ij}^{k} + \text{rand}(-1,1) \times (x_{\max} - x_{\min}). \tag{5}$$

2.3.2　均 衡 项

NMPSO 借鉴 NM 中对信息整合和协调的思想, 充分融合粒子 i 对自身记忆的信息 $\overline{x_{ij}^k} = \sum_{h=1}^{k} x_{ij}^h \Big/ k$、个体本身的局部最优 L_{ij}^k 和群体间的全局最优 G_j^k. 在外部激励下给出均衡项 BL_{ij}^k, 定义为

$$BL_{ij}^k = \theta \times \frac{\overline{x_{ij}^k} + L_{ij}^k + G_j^k}{3}, \tag{6}$$

其中 θ 为外部激励因子, 取 $\theta \in \text{rand}(0,1)$.

2.3.3　改进的速度和位置更新公式

由第 2.3.1 节和第 2.3.2 节得到 NMPSO 算法的速度和位置更新公式为

$$v_{ij}^{k+1} = \lambda_k v_{ij}^k + c_1^k r_1^k (L_{ij}^k - x_{ij}^k) + c_2^k r_2^k (G_j^k - x_{ij}^k) + c_3^k r_3^k (BL_{ij}^k - x_{ij}^k), \tag{7}$$

$$x_{ij}^{k+1} = x_{ij}^k + v_{ij}^{k+1}. \tag{8}$$

为防止粒子远离搜索空间, 粒子每一维的位置都限制在 $[x_{\min}, x_{\max}]$ 之间, 若 $x_{ij}^{k+1} > x_{\max}$, 则 $x_{ij}^{k+1} = x_{\max}$; 若 $x_{ij}^{k+1} < x_{\min}$, 则 $x_{ij}^{k+1} = x_{\min}$; 其余保留原来的位置. 速度修正的原理与上述位置修正的原理一致.

式 (7) 有如下的优势: 1) 从信息系统进化的角度看, 掺入诺兰模型思想的粒子群算法更能平衡局部和全局搜索能力; 2) 当粒子 i 的 x_{ij}^k 与 L_{ij}^k 和 G_j^k 相差很大时, 均衡项的加入会增大 v_{ij}^{k+1}, 从而使粒子的搜索范围扩大, 具有更好的全局搜索能力, 同时收敛速度大幅度提升, 使其更快达到全局最优位置; 3) $\overline{x_{ij}^k}$ 反映了粒子 i 的自身综合能力, 是反映数据集中趋势的一项指标, 即使当 x_{ij}^k 与 L_{ij}^k 和 G_j^k 比较相近时, 由于 θ 的加入, 粒子飞行速度等于 0 的概率相应减少, 从而使粒子更具有多样性; 4) 限定了速度的范围, 有效避免

早熟的发生.

2.3.4 混沌算子设置参数

考虑到Tent混沌映射高搜索效果和均匀分布的特性[25], 为了提高算法对解空间的遍历性, 对λ_{k+1}、c_i^{k+1}和r_i^{k+1}分别使用Tent混沌映射产生, 有

$$\lambda_{k+1}=\begin{cases}2\lambda_k,\ 0\leqslant\lambda_k\leqslant 0.5;\\ 2(1-\lambda_k),\ 0.5<\lambda_k\leqslant 1.\end{cases}\tag{9}$$

$$c_i^{k+1}=\begin{cases}2c_i^k,\ 0\leqslant c_i^k\leqslant 0.5;\\ 2(1-c_i^k),\ 0.5<c_i^k\leqslant 1;\end{cases}$$

$$i=1,2,3.\tag{10}$$

$$r_i^{k+1}=\begin{cases}2r_i^k,\ 0\leqslant r_i^k\leqslant 0.5;\\ 2(1-r_i^k),\ 0.5<r_i^k\leqslant 1;\end{cases}$$

$$i=1,2,3.\tag{11}$$

$$\lambda_0=\mathrm{rand}(0,1),\ c_i^0=\mathrm{rand}(0,1),\ r_i^0=\mathrm{rand}(0,1).\tag{12}$$

由于Tent映射在迭代序列存在小周期, 如0.2、0.4、0.6、0.8; 并且存在不稳定周期点, 如0.25、0.5、0.75, 都可迭代到不动点0, 则对式(9)添加扰动项[27], 使其重新进入混沌状态, 得到

$$\lambda_{k+1}=\begin{cases}2(\lambda_k+10^{-5}\times\mathrm{rand}(\ )),\ 0\leqslant\lambda_k\leqslant 0.5;\\ 2[1-(\lambda_k+10^{-5}\times\mathrm{rand}(\ ))],\ 0.5<\lambda_k\leqslant 1.\end{cases}\tag{13}$$

由于$[\lambda_{\min},\lambda_{\max}]$一般取$[0.4,0.9]$, $[c_{\min},c_{\max}]$一般取$[1.5,2.0]$. 经过Tent混沌映射处理后不一定完全处于该范围内, 需要进行归一化处理, 归一化参数为

$$\lambda_k=0.5\lambda_k+0.4,\tag{14}$$

$$c_i^k=0.5c_i^k+1.5,\ i=1,2,3.\tag{15}$$

2.4 NMPSO算法优化过程

NMPSO算法优化过程如图2所示, 优化步骤如下.

Step 1: 初始化NMPSO算法的控制参数, 包括: 种群规模n, 算法迭代次数m, 惯性因子的取值范围$[\lambda_{\min},\lambda_{\max}]$, 学习因子的取值区间$[c_{\min},c_{\max}]$, 粒子群飞行速度的取值区间$[v_{\min},v_{\max}]$和种群聚集常数$c$.

Step 2: 计算每个粒子当前局部最优L_{ij}^k和当前全局最优G_j^k.

Step 3: 判断收敛条件是否满足, 如果满足, 则转至Step 9.

Step 4: 由式(7)～(11)进行迭代计算, 更新粒子的速度和位置.

Step 5: 计算每个粒子的目标函数值.

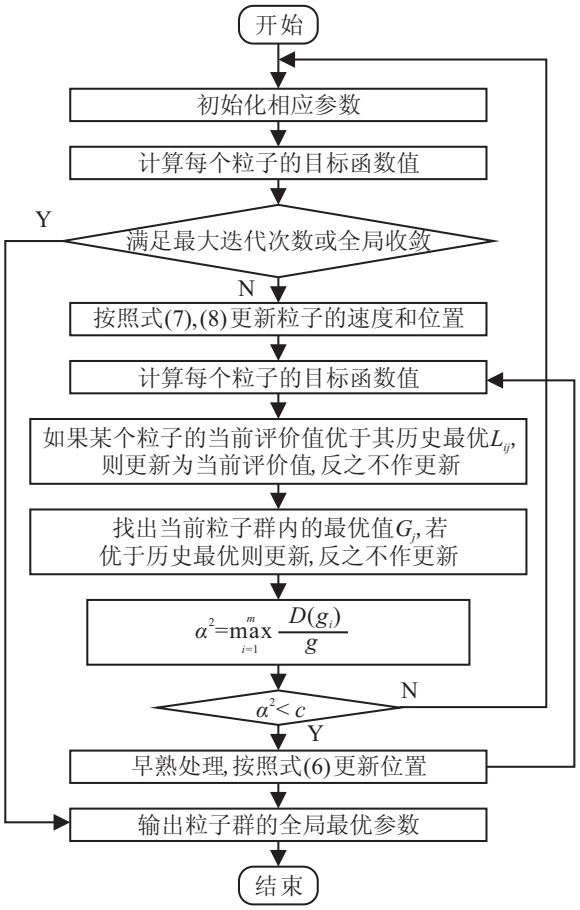

图 2 NMPSO算法的优化流程

Step 6: 比较x_{ij}^k与L_{ij}^k和G_j^k的值, 如果粒子i的当前位置优于其对应的L_{ij}^k和G_j^k的值, 则进行更新.

Step 7: 由式(3)计算群体位置的方差, 并判断$\alpha^2<c$是否成立, 如果成立则按式(5)进行早熟处理, 否则转至Step 3.

Step 8: 由式(5)更新粒子的位置, 转至Step 5.

Step 9: 输出粒子群的最优值, 算法结束.

3 仿真分析

为了评价NMPSO算法在寻优过程中的收敛速度和收敛精度等性能, 选用6个常用的测试函数进行仿真实验, 即

Sphere: $f_1=\sum_{i=1}^{n}x_i^2,\ x_i\in[-1,1]$.

Rosenbrock:

$$f_2=\sum_{i=1}^{n-1}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2],$$

$$x_i\in[-10,10].$$

Rastrigin:

$$f_3=\sum_{i=1}^{n}[x_i^2-10\cos(2\pi x_i)+10],\ x_i\in[-5.12,5.12].$$

Griewank:

$$f_4=1+\sum_{i=1}^{n}\frac{x_i^2}{4\,000}-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right),\ x_i\in[-600,600].$$

Ackley:

$$f_5 = -20e\Big(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\Big) - e\Big(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\Big) + 20 + e,$$

$$x_i \in [-32.768, 32.768].$$

Nonlinear-Rastrigin:

$$f_6 = \sum_{i=1}^{n}[y_i^2 - 10\cos(2\pi y_i) + 10];$$

$$y_i = \begin{cases} x_i, & |x_i| < 0.5; \\ \mathrm{round}(2x_i)/2, & |x_i| \geqslant 0.5; \end{cases}$$

$$x_i \in [-5.12, 5.12].$$

这6个基准函数中, f_2 在 $x = (1, 1, \cdots, 1)$ 处取得全局最小值0, 其余5个都在 $x = (0, 0, \cdots, 0)$ 处取得全局最小值0. f_1 和 f_2 是单峰函数, f_1 常用来测试算法的收敛速度, f_2 是非凸病态单峰函数, 且有局部极小值, 可以用于测试算法能否克服和防止进化中的“早熟”现象; f_3 和 f_4 是多峰函数, f_3 有很多局部极小值, f_4 有多个局部极小值; f_5 是一种可变维数的多峰函数; f_6 是一种非线性函数. $f_1 \sim f_6$ 常被用于检验算法的群体多样性、避免早熟和全局搜索能力的仿真实验中.

本文首先验证在参数取值中增加混沌遍历对算法性能的影响; 然后在基于混沌算子选择的参数上单独与综合评价增加早熟处理和增加均衡项两个改进措施对算法性能的改善; 最后将NMPSO算法与文献[22,28]所提出的算法优化结果进行比较.

3.1　不同参数取值对收敛精度和稳定性的影响

为了验证混沌遍历对算法性能的影响, 选用不同的参数取值方式对算法进行仿真分析. 算法速度更新公式中的参数如下: 随机数 $r_i^k(i = 1, 2, 3)$、惯性因子 λ_k 和认知因子 $c_i^k(i = 1, 2, 3)$. 由于混沌算子的遍历性比随机数更高, 为提高算法中随机数的遍历性, 将式(11)的Tent混沌映射引入到随机数 $r_i^k(i = 1, 2, 3)$ 中, 对 λ_k 和 $c_i^k(i = 1, 2, 3)$ 按不同取值方式(表1)设计4种情形(表2)进行比较验证混沌遍历对算法性能的测试. 测试维数为 $n = 30$, 种群规模为 $N = 200$, 最大迭代次数 $m = 1\,500$, 粒子群的最大速度 $v_{\max}$ 和最小速度 $v_{\min}$ 分别设为测试函数的上限和下限的一半, $\lambda_{\min} = 0.4$, $\lambda_{\max} = 0.9$, 以常数 $c = 40$ 作为早熟收敛判断准则, 分别对6个测试函数进行50次仿真实验, 最后求50次结果的平均值作为评价指标. 表3为NMPSO算法在4种参数取值情况下的优化仿真结果. 其中: MV为函数的平均适应值, OV为算法所找到的函数的最优适应值, Std为最优适应值的标准方差.

表 1　参数的取值方式

参数	取值方式A	取值方式B
λ_k	根据式(9)混沌取值	$\lambda_k = \lambda_{\max} - (\lambda_{\max} - \lambda_{\min})k/m$ $k = 1, 2, \cdots, m$
c_i^k	根据式(10)混沌取值	$c_1^k = c_2^k = c_3^k = 2$ $k = 1, 2, \cdots, m$

表 2　4种情形下参数的取值

	λ_k	c_i^k
情形1	取值方式A	取值方式A
情形2	取值方式B	取值方式B
情形3	取值方式B	取值方式A
情形4	取值方式A	取值方式B

表 3　4种参数取值方式实验结果比较

	f_1			f_2		
	MV	OV	Std	MV	OV	Std
情形1	3.02e-14	0	3.59e-14	3.57e-01	3.70e-03	2.66e-01
情形2	3.11e-14	1.77e-15	5.14e-14	1.74e+01	3.55e-01	4.80
情形3	2.49e-14	6.59e-16	3.21e-14	2.14	3.97e-02	3.60
情形4	1.42e-13	1.64e-15	1.25e-13	1.88e+01	1.85e+01	3.63 e-02
	f_3			f_4		
	MV	OV	Std	MV	OV	Std
情形1	4.37e-11	1.95e-13	4.44e-11	9.57e-10	4.39e-12	7.54e-10
情形2	1.27e-10	3.87e-13	1.45e-10	1.17e-09	1.19e-11	1.19e-09
情形3	2.59e-10	7.53e-13	2.90e-10	1.22e-09	1.09e-11	1.49e-09
情形4	2.60e-09	1.83e-11	2.71e-09	1.20e-08	1.14e-09	1.16e-08
	f_5			f_6		
	MV	OV	Std	MV	OV	Std
情形1	2.79e-06	5.36e-07	1.49e-06	2.14e-10	1.03e-11	2.17e-10
情形2	2.94e-06	4.84e-07	1.41e-06	5.87e-10	1.00e-11	8.07e-10
情形3	2.52e-05	8.37e-07	1.10e-05	1.11e-09	3.26e-11	1.08e-09
情形4	3.86e-06	6.41e-07	1.91e-06	4.13e-10	4.70e-12	5.37e-10

由表3可见, NMPSO算法中, 对所有参数进行混沌取值(情形1)时所有函数所得到的平均适应值(MV)、最优值(OV)和最优适应值的标准方差(Std)都较小, 特别地, NMPSO找到了函数 f_1 的最优值0, 从整体效果综合评价, 按情形1对粒子群算法中的参数进行混沌取值效果最佳.

3.2　算法增加早熟处理和均衡项对性能的影响

为检验本文提出的早熟处理和增加均衡项两个改进措施对算法性能的影响, 采用第3.1节中情形1的参数配置, 选择如下4种组合的算法进行测试: 1) 组合1, 不加早熟, 不加均衡项; 2) 组合2, 只加早熟; 3) 组合3, 只加入均衡项; 4) 组合4, 既加入早熟, 又引进均衡项. 测试结果如表4所示.

对比组合1与组合2、对比组合1与组合3可以看出, 无论是单独加入早熟处理还是单独加入均衡项, 6种测试函数算法的收敛效果都在一定程度上得到了改善, 早熟处理和均衡项均有助于算法的收敛. 综合

表 4 4种情况下算法收敛效果对比

	f_1			f_2		
	MV	OV	Std	MV	OV	Std
情形1	2.78e-02	8.60e-03	1.14e-02	8.46e+01	2.22e+01	5.54e+01
情形2	2.14e-06	7.42e-07	1.24e-06	1.34e+01	1.20e-03	6.63
情形3	2.01e-14	1.92e-15	1.62e-14	1.88e+01	1.85e+01	5.91e-02
情形4	3.02e-14	0	3.59e-14	3.57e-01	3.70e-03	2.66e-01
	f_3			f_4		
	MV	OV	Std	MV	OV	Std
情形1	9.15e+01	5.03e+01	1.95e+01	3.32	1.50	1.18
情形2	1.79e+01	2.08	9.29	4.60e-01	2.38e-01	1.34e-01
情形3	1.65e-10	9.08e-12	1.40e-10	3.06e-10	1.18e-12	2.88e-10
情形4	4.37e-11	1.95e-13	4.44e-11	9.57e-10	4.39e-12	7.54e-10
	f_5			f_6		
	MV	OV	Std	MV	OV	Std
情形1	8.02	5.72	9.90e-01	9.10e+01	4.93e+01	2.14e+01
情形2	4.27	2.57	7.30e+01	3.13e+01	1.05e+01	1.30e+01
情形3	2.97e-06	5.07e-07	1.61e-06	1.39e-10	2.54e-12	1.13e-10
情形4	2.79e-06	5.36e-07	1.49e-06	2.14e-10	1.03e-11	2.17e-10

比较4种组合, 加入早熟处理和均衡项两种改进措施的组合4对于测试函数 f_2、f_3、f_5 寻优效果最好; 对于函数 f_1、f_6, 平均适应值(MV)略低于组合3, 但是所能找到的最优值(OV)却远好于其他组合, 尤其对于函数 f_1 能找到最优值0, 在收敛精度上得到了很大的提升.

3.3 算法优化结果的比较

蚂蚁群(ASO)算法和PSO算法同属于优化算法, 并且在算法上对信息的交流有一定的相似性. 本文选择文献[27]中的混沌蚂蚁群优化(CASO)算法和带创造性思维的混沌蚂蚁群优化算法(CTCASO)与本文算法进行比较. 表5为3种算法各基准函数独立运行50次优化求解的结果比较(其中 $n = 30$, $N = 200$, $m = 1\,500$).

表 5 与不同ASO算法所得平均适应值的实验结果比较

	f_1	f_2	f_3	f_4	f_5	f_6
CASO[27]	3.81e-03	2.34e+01	2.26e+01	4.66e-01	3.55e-06	—
CTCASO[27]	3.35e-11	2.29e+01	1.58e+00	2.51e-10	9.17e-01	—
NMPSO	3.02e10-14	3.57e-01	4.37e-11	9.57e-10	2.79e-06	2.14e-10

由表5平均适应值的比较结果可见, NMPSO算法对各函数的求解质量相对较高, 除了测试Griewank函数时的CTCASO算法优于NMPSO的相应值外, 其余的平均适应值在利用NMPSO算法优化后都有不同数量级的提高, 其中最显著的NMPSO算法与CTCASO算法比较, Rastrigin函数的平均适应值提高了11个数量级. 另外, 分别采用基本粒子群算法(BPSO)[1-2]、标准粒子群(SPSO)算法[22]、全信息粒子群(FIPS)算法[28]和本文所提出的NMPSO进行50次对比寻优测试(其中 $n = 30$, $N = 200$, $m = 1\,500$), 收敛效果如图3所示, 测试结果如表6所示.

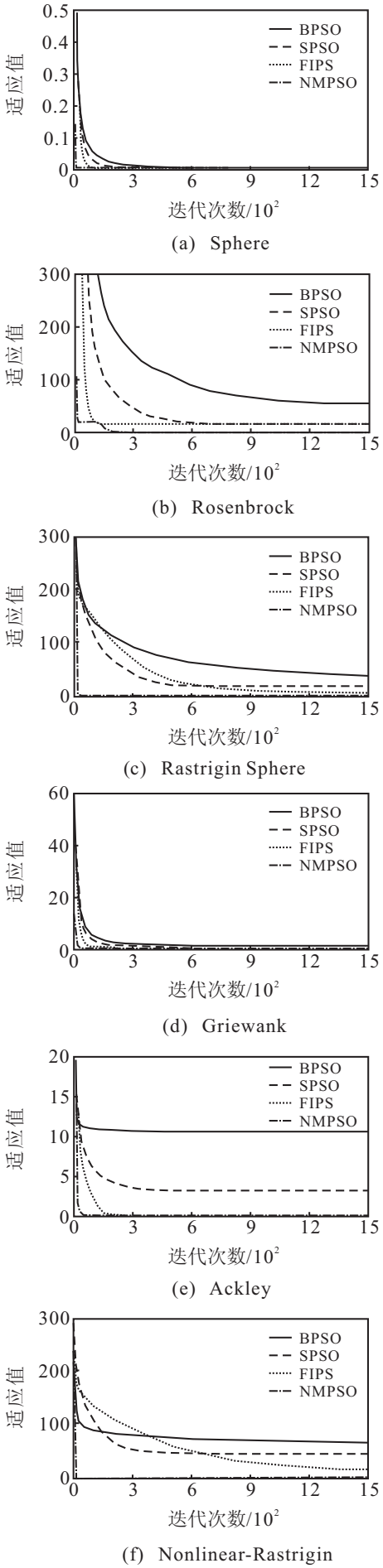

图 3 各函数4种PSO对比曲线

表 6　与不同PSO算法所得平均适应值结果比较

	f_1	f_2	f_3	f_4	f_5	f_6
BPSO[1-2]	3.89e-04	5.39e+01	3.91e+01	1.03	1.05e+01	6.80e+01
SPSO[22]	3.81e-06	1.70e+01	1.96e+01	7.00e-02	3.17	4.55e+01
FIPS[28]	3.81e-06	1.70e+01	1.96e+01	7.47e-04	5.79e-15	1.55e+01
NMPSO	3.02e-14	3.57e-01	4.37e-11	9.57e-10	2.79e-06	2.14e-10

由表6可知, 除了 f_1 和 f_5 外, NMPSO算法对其他函数寻优得到的平均适应值均比BPSO、SPSO和FIPS有不同数量级的提高. 但由图3可见, NMPSO相对于另外3种算法而言, 对于所有测试函数算法收敛速度明显加快. 因此, NMPSO算法在保证寻优精度的情况下, 寻优速度较其他方法更快, 将诺兰模型引入粒子群算法有效地提高了算法的全局和局部收敛能力.

为了进一步表明NMPSO算法性能的优势, 运用双侧 T- 检测法将NMPSO与BPSO、SPSO、FIPS分别根据式(16)和(17)进行显著性检验, 有

$$s_{12}^2 = \frac{(n_1-1)s_1^2 + (n_2-1)s_2^2}{n_1+n_2-2}, \tag{16}$$

$$t = \frac{\overline{D_1} - \overline{D_2}}{s_{12}\sqrt{\dfrac{1}{n_1}+\dfrac{1}{n_2}}}. \tag{17}$$

其中: n_1、n_2 为样本数, s_1、s_2 为标准差, $\overline{D_1}$、$\overline{D_2}$ 为样本均值, s_{12} 为差数平均的标准差.

每次测试独立运行50次, 两个样本对比的检验自由度为 $d_f = n_1-1+n_2-1 = 98$. 取显著性水平 $\alpha = 0.01$, 查表可知 $t_{0.01/2} = t_{0.005} = 2.626$. 当 $|t| < t_{0.005}$ 时, 认为算法NMPSO与比较算法无显著性差异, 反之认为差异显著. 对于函数 f_2、f_3、f_4、f_6, NMPSO相比其他算法求得的 t 值的绝对值均大于 $t_{0.005}$, 因此表明NMPSO在 f_2、f_3、f_4、f_6 函数寻优问题上改进显著.

4　结　论

本文从系统的认知分析过程和角度出发, 分析了PSO算法的基本原理, 将诺兰模型的6个阶段引入PSO算法中, 提出了基于诺兰模型思想的PSO算法(NMPSO), 利用粒子多方面信息给出了均衡项改进速度更新公式. 运用混沌思想提高算法搜索的遍历性, 利用粒子欧氏距离指数提高粒子群多样性. 通过对该算法自身参数的不同取值的探讨发现, 对粒子群的各参数进行混沌优化, 收敛结果总体上最好. 将所提出的NMPSO算法与其他文献中的改进PSO和CASO算法进行比较, 仿真结果表明, 所提出算法平均求解的质量相对较高. 下一步将针对PSO算法尚存的收敛速度和精度上的缺点进行更加深入的理论研究, 希望借助粒子的运动系统和粒子的转移状态, 探索出新的改进PSO算法, 最终实现加快算法收敛速度和精度的目的, 并将其应用到求解实际系统的约束优化问题上.

参考文献(References)

[1] Eberhart R, Kennedy J. A new optimizer using particle swarm theoy[C]. Proc of the 16th Int Symposium on Micro Machine and Human Science. Piscataway: IEEE Seervice Center, 1995: 39-43.

[2] Kennedy J, Eberhart R. Particle swarm optimization[C]. IEEE Int Conf on Networks. Perth: IEEE Press, 1995: 1942-1948.

[3] Yin P Y, Yu S S, Wang P P, et al. Multi-objective task allocation in distributed computing systems by hybrid particle swarm optimization[J]. Applied Mathematics and Computation, 2007, 184(2): 407-420.

[4] del Valle Y, Venayagamoorthy G, Mohagheghi S, et al. Particle swarm optimization: Basic concepts, variants and applications in power systems[J]. IEEE Trans on Evolutionary Computation, 2008, 12(2): 171-195.

[5] Rainer Storn. Designing nonstandard filters with differential evolution[J]. IEEE Signal Processing Magazine, 2005, 22(1): 103-106.

[6] Park J, Choi K, Allstot D J. Parasitic-aware RF circuit design and optimization[J]. IEEE Trans on Circuits and Systems I: Fundamental Theory and Applications, 2004, 51(10): 1953-1966.

[7] Mukherjee V, Ghoshal S P. Intelligent particle swarm optimized fuzzy PID controller for AVR system[J]. Electric Power Systems Research, 2006, 77(12): 1689-1698.

[8] Wu H, Sun F C, Sun Z Q, et al. Optimal trajectory planning of a flexible dual-arm space robot with vibration reduction[J]. J of Intelligent & Robotic Systems, 2004, 40(2): 147-163.

[9] Pavlidis N G, Parsopoulos K E, Vrahatis M N. Computing nash equilibria through computational intelligence methods[J]. J of Computational and Applied Mathematics, 2005, 175(1): 113-136.

[10] Rasmussen T K, Krink T. Improved hidden Markov model training for multiple sequence alignment by a particle swarm optimization-evolutionary algorithm hybrid[J]. Biosystems, 2003, 72(1/2): 5-17.

[11] Shen Q, Jiang J H, Jiao C X, et al. Modified particle swarm optimization algorithm for variable selection in mlr and pls modeling: Qsar studies of antagonism of angiotensin ii antagonists[J]. European J of Pharmaceutical Sciences, 2004, 22(2/3): 145-152.

[12] 沈林成, 霍霄华, 牛轶峰. 离散粒子群优化算法研究现状综述[J]. 系统工程与电子技术, 2008, 30(10): 1986-1990.

(Shen L C, Huo X H, Niu Y F. Survey of discrete particle swarm optimization algorithm[J]. Systems Engineering and Electronics, 2008, 30(10): 1986-1990.)

[13] Gregorio Toscano Pulido, Carlos A Coello Coello. A constraint-handling mechanism for particle swarm optimization[C]. IEEE CEC06. Portland: IEEE, 2004: 1396-1403.

[14] Margarita Reyes-Sierra, Carlos A Coello Coello. Multi-objective particle swarm optimizers: a survey of the state-of-the-art[J]. Int J of Computational Intelligence Research, 2006, 2(3): 287-308.

[15] Carlisle A, Dozierv G. Adapting particle swarm optimization to dynamic environments[C]. Proc of the Int Conf on Artificial Intelligence. Las Vegas: CSREA Press, 2000: 429-434.

[16] 齐名军, 杨爱红. 一种混沌优化机制的双量子粒子群优化算法[J]. 计算机工程与应用, 2009, 45(30): 34-36.
(Qi M J, Yang A H. Double quantum delta particle swarm optimization based on chaos optimization strategy[J]. Computer Engineering and Applications, 2009, 45(30): 34-36.)

[17] 刘道华, 原思聪, 兰洋, 等. 混沌映射的粒子群优化设计[J]. 西安电子科技大学学报, 2010, 37(4): 764-768.
(Liu D H, Yuan S C, Lan Y, et al. Method of particle swarm optimization based on the chaos map[J]. J of Xidian University, 2010, 37(4): 764-768.)

[18] 匡芳君, 徐蔚鸿, 张思扬. 基于改进混沌粒子群的混合核SVM参数优化及应用[J]. 计算机应用研究, 2013, 31(3): 671-674.
(Kuang F J, Xu W H, Zhang S Y. Parameter optimization and application of SVM with mixtures kernels based on improved chaotic particle swarm optimization[J]. Application Research of Computers, 2013, 31(3): 671-674.)

[19] 唐苏妍, 朱一凡, 张伟, 等. 一种基于创造性思维的粒子群优化算法[J]. 控制与决策, 2011, 26(8): 1181-1186.
(Tang S Y, Zhu Y F, Zhang W, et al. Particle swarm optimization algorithm based on creative thinking[J]. Control and Decision, 2011, 26(8): 1181-1186.)

[20] 赵辉. 诺兰模型的启示[Z]. 中国计算机报, 2007-5-28(B16).
(Zhao H. The enlightenment of nolan model[Z]. China Information Word, 2007-5-28(B16).)

[21] 韩崇昭, 朱洪艳, 段战胜. 多源信息融合[M]. 第2版. 北京: 清华大学出版社, 2010: 1-25.
(Han C Z, Zhu H Y, Duan Z S. Multi-source information fusion[M]. 2nd ed. Beijing: Tsinghua University Press, 2010: 1-25.)

[22] Shi Y, Eberhart R. A modified particle swarm optimization[C]. Proc of the IEEE Congress on Evolutionary Computation. Piscataway: IEEE Press, 1998: 303-308.

[23] Shi Y, Eberhart R. Empirical study of particle swarm optimization[C]. Proc of the Congress on Evolutionary Computation. Washington DC: IEEE, 1999: 1945-1950.

[24] Chen X, Li Y. An improved stochastic PSO with high exploration ability[C]. Proc of the IEEE Congress on Swarm Intellgence Symposium. Indianapolis: IEEE, 2006: 228-235.

[25] 赵欣. 不同一维混沌映射的优化性能比较研究[J]. 计算机应用研究, 2012, 29(3): 913-915.
(Zhao X. Research on optimization performance comparison of different one-dimensional chaotic maps[J]. Application Research of Computers, 2012, 29(3): 913-915.)

[26] 张浩, 张铁男, 沈继红, 等. Tent混沌粒子群算法及其在结构优化决策中的应用[J]. 控制与决策, 2008, 23(8): 857-862.
(Zhang H, Zhang T N, Shen J H, et al. Research on decision-makings of structure optimization based on improved Tent PSO[J]. Control and Decision, 2008, 23(8): 857-862.)

[27] 李玉英. 带创造性思维的混沌蚂蚁优化算法[J]. 控制与决策, 2014, 29(5): 937-940.
(Li Y Y. Chaotic ant swarm optimization algorithm with creative thinking[J]. Control and Decision, 2014, 29(5): 937-940.)

[28] Mendes R, Kennedy J, Neves J. The fully informed particle swarm: Simpler, maybe better[J]. IEEE Trans on Evolutionary Computation, 2004, 8(3): 204-210.

(责任编辑：郑晓蕾)