文章编号: 1001-0920(2004)11-1298-03

一种带有梯度加速的粒子群算法

王俊伟, 汪定伟
(东北大学 信息科学与工程学院, 辽宁 沈阳 110004)

摘 要: 通过引入梯度信息来影响粒子速度的更新, 构造了一种带有梯度加速的粒子群算法. 为减小陷入局优的可能性, 当群体最优信息陷入停滞时, 对群体进行部分初始化来保持群体的活性, 并讨论了改进算法的适用范围. 仿真结果表明, 对于单峰函数和多峰函数, 改进算法都能够取得较好的优化效果.

关键词: 粒子群算法; 演化计算; 随机搜索
中图分类号: TP18 **文献标识码**: A

Particle swarm optimization algorithm with gradient acceleration

WANG Jun-wei, WANG Ding-wei
(School of Information Science and Engineering, Northeastern University, Shenyang 110004, China. Correspondent: WANG Ding-wei; E-mail: dwwang@mail.neu.edu.cn)

Abstract: By adding gradient information to influence the update of velocities of the particles, a kind of particle swarm optimization (PSO) algorithm with gradient acceleration is proposed. When the optimum information of the swarm is stagnant, some particles in the population are initialized again to reduce the possibility of trapping in local optimum. The scope of application is also discussed, and the result of computer simulation indicates that the improved PSO could get better performance in one-peak functions and multi-peak functions.

Key words: particle swarm optimization; evolutionary computation; stochastic searching

1 引 言

粒子群算法(PSO)是 Kennedy 和 Eberhart 于 1995 年提出的一种演化算法[1], 它源于对简化社会模型的模拟. PSO 算法最早用于函数优化和神经元网络的训练, 其后又应用于其他领域, 都取得了较好的效果. 目前, 这一算法已引起人们的普遍关注, 并已出现了许多研究成果[2]. 标准 PSO 算法具有很强的通用性, 适用于各种优化问题, 特别是实优化问题. 然而, 这种广泛的适应性通常会导致对具体问题的特性考虑不够, 比如没有考虑问题中有效的梯度信息, 后期迭代搜索效率不高等[3].

针对以上问题, 本文提出一种带有梯度加速的 PSO 算法, 对标准 PSO 算法进行改进. 在粒子的速度更新中, 以一定概率加入梯度信息, 使粒子的移动更具针对性和效率. 为减小粒子陷入局优的可能性, 对群体最优值进行观察. 在寻优过程中, 当最优信息出现停滞时, 对部分粒子进行重新初始化, 从而保持群体的活性. 最后通过实验对带有梯度加速的 PSO 算法的适用范围进行讨论. 仿真实验表明, 对于单峰函数和多峰函数, 改进算法都能够取得较好的优化效果.

2 标准 PSO 算法

由 m 个粒子(Particle)组成的群体(Swarm)对 D 维搜索空间进行搜索. 每个粒子在搜索时, 考虑了自己搜索到的历史最好点和群体内(或邻域内)所有粒子的历史最好点; 在此基础上进行位置(状态, 也就是解)的变化. 第 i 个粒子的位置和速度分别表示为

收稿日期: 2003-08-25; **修回日期**: 2003-10-13.

作者简介: 王俊伟(1979—), 男, 辽宁沈阳人, 博士生, 从事复杂系统建模与优化的研究; 汪定伟(1948—), 男, 江西彭泽人, 教授, 博士生导师, 从事生产计划与调度、建模与决策等研究.

$$x_i = (x_{i1}, x_{i2}, \cdots, x_{iD}),$$
$$v_i = (v_{i1}, v_{i2}, \cdots, v_{iD}),$$
$$1 \leq i \leq m, 1 \leq d \leq D; \quad (1)$$

它所经历过的历史最好点表示为

$$p_i = (p_{i1}, p_{i2}, \cdots, p_{iD}); \quad (2)$$

群体内(或邻域内) 所有粒子所经过的最好点表示为

$$p_g = (p_{g1}, p_{g2}, \cdots, p_{gD}). \quad (3)$$

粒子的位置和速度根据如下方程进行变化:

$$v_{id}^{k+1} = \omega v_{id}^{k} + c_1 \xi (p_{id}^{k} - x_{id}^{k}) + c_2 \eta (p_{gd}^{k} - x_{id}^{k}), \quad (4)$$

$$x_{id}^{k+1} = x_{id}^{k} + v_{id}^{k+1}. \quad (5)$$

其中: ω为惯性权重, 它决定了对粒子当前速度继承的多少, 选择合适可使粒子具有均衡的探索和开发能力; c_1 和 c_2 是学习因子, 为正常数, 通常取值为 2(文献中也有其他的取值); $\xi, \eta \in U[0, 1]$ 是在 $[0, 1]$ 区间内均匀分布的伪随机数

算法的流程与遗传算法等进化算法相似, 经历初始化、适应值评价、速度和位置的更新、停止准则比较等步骤, 这里不再详述

3　带有梯度加速的 PSO 算法

与其他进化算法一样, 标准 PSO 算法利用种群进行随机搜索, 没有考虑具体问题的特性, 不使用梯度信息, 而梯度信息往往包含目标函数的一些重要信息

对于函数 $f(x)$, $x = (x_1, x_2, \cdots, x_n)$, 其梯度可表示为

$$\nabla f(x) = \left[\frac{\partial f(x)}{\partial x_1}, \frac{\partial f(x)}{\partial x_2}, \cdots, \frac{\partial f(x)}{\partial x_n} \right]^{\mathrm{T}}, \quad (6)$$

函数值的最速下降方向是负梯度方向

本文通过引入梯度信息来影响粒子速度的更新, 构造出一种带有梯度加速的 PSO 算法 每次粒子进行速度和位置更新时, 每个粒子以概率 p 按式 (1) 和式(2) 更新, 并以概率(1 - p) 按梯度信息更新, 在负梯度方向进行一次直线搜索来确定移动步长 直线搜索采用黄金分割法 为防止陷入局优, 在群体最优信息陷入停滞时, 对群体进行部分重新初始化, 以保持群体的活性, 减小群体陷入局优的可能性

梯度信息的加入使粒子的移动更具针对性和效率, 进一步提高了 PSO 算法的收敛速度, 但也会增加算法对问题的依赖性, 特别是有些梯度信息极易将粒子引入局优 所以带有梯度加速的 PSO 算法需要根据问题的性质来调整梯度信息对于粒子移动的影响程度

带有梯度加速的 PSO 算法主要步骤如下:

1) 在初值范围内随机初始化粒子种群, 包括随机位置和速度

2) 评价每个粒子的适应值

3) 将每个粒子的适应值与其经历过的最好值进行比较, 如果更好, 则将其作为当前粒子的个体最优值

4) 将每个粒子的个体最优值与群体最优值进行比较, 如果更好, 则将其作为群体最优值; 若规定迭代次数内群体最优值未得到更新, 则按一定概率将部分粒子重新初始化

5) 对于每个粒子的速度和位置, 以概率 p 按式 (1) 和式(2) 更新, 并以概率(1 - p) 按梯度信息更新, 在负梯度方向进行一次直线搜索来确定移动步长

6) 若未达到终止条件, 则转步骤 2).

4　实验与分析

为比较标准 PSO 算法和带有梯度加速的 PSO 算法, 并研究改进后算法对问题的依赖性, 本文选取常用的 5 个测试函数[4, 5] 进行实验(见表 1), 它们的最优值均为 0. 算法用 Java 语言实现, 在 PⅢ-1G 的 Dell 计算机上运行.

标准 PSO 算法及带有梯度加速的 PSO 算法分别选取种群 30 和种群 10, 并选取不同的惯性权重(这里均采用时变权重). 为全面比较优化效果, 采用两种停止准则: 1) 是否达到规定的达优值(观察其达优率、平均迭代次数及平均计算时间); 2) 迭代 4 000 次, 观察其取得的最优值

在改进 PSO 算法中, 当 20 次迭代未取得更好的群体最优值时, 认为群体最优信息陷入停滞, 在粒子群体中随机选取 30% 重新初始化, 重新初始化的方式为在初值范围内随机产生一个新的粒子代替原粒子. 对于第个粒子信息的速度和位置以概率 1% 按照梯度信息进行更新, 种群 30 下的优化结果如表 2 所示

函数 2 和函数 3 为单峰函数, 改进 PSO 算法对它们的优化效果优于标准 PSO 算法; 函数 1、函数 4 和函数 5 是多峰函数, 改进 PSO 算法也可取得较好的优化效果 特别是在不同的惯性权重设置下, 都能取得良好的达优率, 迭代 4 000 次所取得的最优值比标准 PSO 也有很大的改进 改进算法每次迭代计算时间要长于标准 PSO 算法, 但由于达优率的提高, 对于函数1～函数3和函数5, 平均运行时间

表 1 标准 PSO 和带有梯度加速的 PSO 算法比较

序号	函数名	表 达 式	维数 n	初值范围	达优值
函数 1	Schaffer's f_6	$f_6(x) = 0.5 - \frac{(\sin\sqrt{x_1^2 + x_2^2})^2 - 0.5}{(1 + 0.001(x_1^2 + x_2^2))^2}$	2	$[-100, 100]^2$	10^{-5}
函数 2	Sphere	$f_2(x) = \sum_{i=1}^{n} x_i^2$	30	$[-100, 100]^n$	0.01
函数 3	Rosenbrock	$f_3(x) = \sum_{i=1}^{n-1} (100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2)$	30	$[-30, 30]^n$	100
函数 4	Rastrigrin	$f_4(x) = \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i) + 10)$	30	$[-5.12, 5.12]^n$	100
函数 5	Griewank	$f_5(x) = \frac{1}{4\,000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\frac{x_i}{\sqrt{i}} + 1$	30	$[-600, 600]^n$	0.1

表 2 种群 30 下的优化结果

惯性权重	指 标	函数 1		函数 2		函数 3		函数 4		函数 5	
		SPSO	IPSO	SPSO	IPSO	SPSO	IPSO	SPSO	IPSO	SPSO	IPSO
0.9～0.2	达优率	0.88	1	1	1	0.77	0.97	1	1	1	1
	迭代次数	730.72	476.40	798.28	4.31	939.29	965.07	598.83	706.49	763.14	4.47
	运行时间	0.61	0.28	0.45	0.02	1.06	0.69	0.53	0.74	0.69	0.02
0.5～0.2	达优率	0.69	1	1	1	0.91	0.98	1	1	1	1
	迭代次数	496.81	165.67	389.2	4.96	655.85	430.70	197.33	2 673.18	370.8	4.80
	运行时间	0.86	0.10	0.28	0.02	0.55	0.32	0.17	0.27	0.39	0.02
0.9～0.5	达优率	0.97	1	1	1	0.88	0.88	1	1	1	1
	迭代次数	686.66	677.90	1 165.21	4.96	1 541.28	1 380.47	898.64	1 035.61	1 118.99	5.10
	运行时间	0.43	0.39	0.63	0.01	1.04	1.08	0.78	1.07	1.06	0.02
迭 代 4 000 次	达优率	0.98	1	1	1	0.87	0.97	1	1	1	1
	平均最优值	1.94E-4	0.0	7.04E-19	3.62E-35	53.0	21.2	36.08	10.0	0.013 9	1.15E-16

注: SPSO 为标准 PSO, IPSO 为改进 PSO; 惯性权重标明的是时变权重取值范围; 递减率为变化范围/1 000; 迭代次数为平均迭代次数; 运行时间为平均运行时间, 单位为 s.

要小于标准 PSO 算法(函数 2 效果最好). 函数 4 是一个复杂的多峰函数, 梯度加速的 PSO 算法迭代 4 000 次可以取得更好的最优值, 但平均运行时间稍有增加

在种群 10 下上述结论仍然成立, 并且表现得更加明显, 改进的 PSO 算法仍有很高的达优率, 并且种群的减小使计算时间大幅度缩短 由于篇幅所限, 种群 10 下的优化结果略

由上述结果可以看出, 带有梯度加速的 PSO 算法具有一定的依赖性, 改进的 PSO 算法适用于单峰函数和一般的多峰函数 梯度信息可使粒子的搜索更有效, 使改进算法的种群依赖性减小, 而对复杂多峰函数的优化效果则不理想

5 结 论

本文将梯度信息引入标准 PSO 算法, 并在群体最优信息陷入停滞时对群体进行部分初始化, 以此来保持群体的活性, 防止群体陷入局优, 构造出带有梯度加速的 PSO 算法 通过实验讨论了改进算法的适用范围 实验表明, 对于单峰函数和多峰函数, 带有梯度加速的 PSO 都能取得较好的优化效果 对于复杂的多峰函数, 有时梯度信息的使用会使算法的平均运行时间稍有增加

(下转第 1304 页)

$$y_m(k+1) = 0.6y_m(k) + 0.2y_m(k-1) + r(k) = AX_m(k) + br(k).$$

式中: $r(k) = \sin(2\pi k/25)$, $A = [0.6, 0.2]$, $X_m(k) = [y_m(k) \quad y_m(k-1)]^T$, $b = 1$.

根据定理 1 设计形如式(6)的自适应控制器, 反馈增益 $\rho = 0.35$. 例中 GFHM 形如式(3), 辨识器初值为

$$p = 1.7966,$$
$$q = [q_1, q_2, q_3] = [-0.8234, 2.8879, 0.9175],$$
$$x = [x_1 \quad x_2 \quad x_3]^T = [y(k) + 0.0135 \quad y(k) - 1.3630 \quad y(k-1) + 0.6249]^T,$$
$$K_x = \mathrm{diag}[k_1, k_2, k_3] = \mathrm{diag}[-0.5893, 0.2256, -1.9538].$$

仿真结果如图 2 和图 3 所示, 其中图 2(a) 和图 3(a) 中的实线 y 分别为本文和文献[6]中的控制器输出曲线; 虚线 y_m 均为参考模型输出曲线. 通过对比图 2 和图 3 可见, 本文的控制器跟踪效果明显好于文献[6].

5 结 语

本文提出一种基于 GFHM 的自适应控制器, 并利用 Lyapunov 稳定性理论证明了此控制器是全局渐近稳定的, 跟踪误差趋于零. 该方法具有以下特点: 1) 辨识参数少, 辨识复杂性较小; 2) 克服了在模型辨识精度要求较高时, 规则爆炸所带来的无法在线辨识的问题, 可以用神经网络在线优化参数; 3) 控制器的跟踪效果优于文献[6], 控制精度较高.

参考文献(References):

[1] Tseng C S, Chen B S, Uang H J. Fuzzy tracking control design for nonlinear dynamic systems via T-S fuzzy model[J]. *IEEE Trans on Fuzzy System*, 2001, 9(3): 381-392.

[2] Zhang H G, Quan Y B. Modeling, identification and control of a class of nonlinear system [J]. *IEEE Trans on Fuzzy System*, 2001, 9(2): 349-354.

[3] 全永兵, 张化光. 广义模糊双曲正切模型及其逼近性研究[J]. 东北大学学报, 2003, 24(1): 1-3.
(Quan Y B, Zhang H G. Generalized fuzzy hyperbolic model: A universal approxomator [J]. *J of Northeastern University*, 2003, 24(1): 1-3.)

[4] 张化光, 王智良, 黎明, 等. 广义模糊双曲正切模型: 一个万能逼近器[J]. 自动化学报, 2004, 30(3): 416-422.
(Zhang H G, Wang Z L, Li M, et al. Generalized fuzzy hyperbolic model: A universal approximator [J]. *Acta Automatic Asinica*, 2004, 30(3): 416-422.)

[5] 陈新海, 李严俊, 周军. 自适应控制系统及应用[M]. 西安: 西北工业大学出版社, 1998.

[6] Wang L X. *Adaptive Fuzzy Systems and Control: Design and Stability Analysis* [M]. NJ: Prentice Hall, 1994.

(上接第 1300 页)

参考文献(References):

[1] Kennedy J, Eberhart R C. Particle swarm optimization [A]. *IEEE Int Conf on Neural Networks*[C]. Perth, 1995. 1942-1948.

[2] 谢晓锋, 张文俊, 杨之廉. 微粒群算法综述[J]. 控制与决策, 2003, 18(2): 129-133.
(Xie X F, Zhang W J, Zhang Z L. Overview of particle swarm optimization [J]. *Control and Decision*, 2003, 18(2): 129-133.)

[3] Angeline P J. Evolutionary optimization versus particle swarm optimization: Philosophy and performance difference [A]. *Proc of 7th Annual Conf on Evolutionary Programming* [C]. Germany, 1998. 601-610.

[4] Clerc M, Kennedy J. The particle swarm: Explosion, stability and convergence in a multi-dimensional complex space [J]. *IEEE J of Evolutionary Computation*, 2001, 6(1): 58-72.

[5] Trelea I C. The particle swarm optimization algorithm: Convergence analysis and parameter selection [J]. *Information Processing Letters*, 2003, 85: 317-325.

[6] Kennedy J. The particle swarm: Social adaptation of knowledge [A]. *IEEE Int Conf on Evolutionary Computation* [C]. Indianapolis, 1997. 303-308.

[7] Eberhart R C, Shi Y. Comparing inertia weigthts and constriction factors in particle swarm optimization [A]. *Proc of the IEEE Congress on Evolutionary Computation* [C]. San Diego, 2000. 84-88.

[8] Shi Y, Eberhart R C. Parameter selection in particle swarm optimization [A]. *Proc of the Seventh Annual Conf on Evolutionary Programming* [C]. New York, 1998. 591-600.