第 24 卷 第 10 期 控 制 与 决 策 2009 年 10 月
Vol. 24 No. 10 Control and Decision Oct. 2009

文章编号：1001-0920(2009)10-1513-04

一种动态分级的混合粒子群优化算法

龙 文，梁昔明，肖金红，阎 纲
（中南大学 信息科学与工程学院，长沙 410083）

摘 要：针对粒子群算法早熟收敛和搜索精度不高的问题，提出一种动态分级的混合粒子群优化算法. 该算法采取 3 种级别的并行粒子群算法，分别用于全局搜索和局部搜索及二者的结合，并根据搜索阶段动态调整各种级别中并行变量的数目. 在全局搜索中，将混沌机制引入算法中以增强算法的全局搜索能力；在局部搜索中，采用单纯形法对适应度最优解进行局部寻优. 仿真实验表明，该算法比其他优化算法具有更好的性能.

关键词：动态分级；粒子群优化算法；混沌；单纯形法
中图分类号：TP18 **文献标识码**：A

Dynamic hierarchical hybrid particle swarm optimization algorithm

LONG Wen, LIANG Xi-ming, XIAO Jin-hong, YAN Gang
(School of Information Science and Engineering, Central South University, Changsha 410083, China. Correspondent: LONG Wen, E-mail: lw770457@163. com)

Abstract: A dynamic hierarchical hybrid particle swarm optimization (DHHPSO) algorithm is proposed for the problem of the premature and low precision of the standard PSO. In the DHHPSO algorithm, by using parallel PSO algorihtm, hierarchical parallel variables are employed for global and local search respectively. Hierarchical ways of parallel variables are dynamically adapted according to the search phases. In the global search, chaotic mechanism is introduced to the algorithm to enhance the global search capability. In the local search, simplex method is used to search local optimization solution. Simulations show that this algorithm has better optimization performance than other global algorithms.

Key words: Dynamic hierarchical; Particle swarm optimization algorithm; Chaos; Simplex search

1 引 言

粒子群优化算法[1]（PSO）是由 Kennedy 博士等人于 1995 年提出的一种模拟鸟类捕食行为的全局优化算法. 该算法的前期收敛速度快、设置参数少、容易实现、能有效地解决复杂优化问题，在函数优化、模式识别、机器人学习、组合优化以及一些工程领域得到了广泛应用. 但粒子群算法和其他全局优化算法一样，有易陷于局部最优、早熟收敛的缺点. 如何使全局优化算法避免出现早熟收敛，一直是众多研究者关注的重点. 目前出现了许多改进算法，改进算法主要是针对惯性权重因子 w 和保持种群多样性进行的，如：文献[2]研究了惯性权重因子 w 对优化性能的影响，发现较大的 w 值有利于跳出局部极值点，而较小的 w 值有利于算法的收敛；文献[3]则通过对粒子位置或速度引入一个小概率随机变异操作来增强种群的多样性，使算法能有效地进行全局搜索.

大量研究实验表明，克服全局优化算法早熟收敛的措施[4-8]主要有：设法保持种群的多样性，或引入跳出局部最优点的机制，或与其他算法融合. 鉴于此，本文在文献[9]的基础上提出一种新的混合粒子群优化算法. 该算法采用 3 种级别的并行粒子群算法，分别用于全局搜索和局部搜索以及二者的结合，并根据搜索阶段动态调整各种级别中并行变量的数目. 在全局搜索中，将混沌机制引入到算法中以增强算法的全局搜索能力；在局部搜索中，采用单纯形法对适应度最优解进行局部寻优. 动态分级的混合粒子群优化（DHHPSO）算法具有粒子群优化的全局搜索能力，动态分级实现了并行变量之间的协同配合、信息交换，提高了优化效率和搜索精度. 仿真实验表明，该算法比其他全局优化算法具有更好的性

收稿日期：2008-11-19；**修回日期**：2009-02-12.
基金项目：国家自然科学基金项目(60874070)；高等学校博士点基金项目(20070533131).
作者简介：龙文(1977—)，男，湖南隆回人，博士生，从事智能优化方法及应用、过程控制及系统优化的研究；
梁昔明(1967—)，男，湖南汨罗人，教授，博士生导师，从事过程控制及系统优化、进化计算的研究.

能.

2 标准粒子群优化算法

下面应用 PSO 算法模拟鸟群的捕食行为. 设想这样一个场景:一群鸟在随机搜索食物,该区域内只有一块食物,所有的鸟都不知道食物在哪,但它们知道当前的位置离食物还有多远,那么找到食物的最优策略是什么呢?最简单有效的就是搜索目前离食物最近的鸟的周围区域. PSO 从该模型中得到启示并用于解决优化问题. PSO 中每个优化问题的解都是搜索空间中的一只鸟,称之为“粒子”. 所有粒子都有一个由被优化函数决定的适应值(候选解)和一个决定它们飞翔方向与距离的速度. 在优化过程中,每个粒子记忆、追随当前的最优粒子,在解空间中进行搜索. PSO 算法的初始化为一群随机粒子(随机候选解),然后通过迭代找到最优解. 在每一次迭代过程中,粒子通过追逐两个极值更新自己的位置:一个是粒子自身所找到的当前最优解,称其为个体极值 p_i;另一个是整个群体当前找到的最优解,称其为全局极值 p_g.

假设用 $X_i=(x_{i1},x_{i2},\cdots,x_{iD})^{\mathrm{T}}$ 表示第 i 个粒子,其中 D 为粒子的维数;最好位置为 $P_i=(p_{i1},p_{i2},\cdots,p_{iD})^{\mathrm{T}}$;整个群体的最好位置为 $P_g=(p_{g1},p_{g2},\cdots,p_{gD})^{\mathrm{T}}$;粒子 i 的速度为 $V_i=(v_{i1},v_{i2},\cdots,v_{iD})^{\mathrm{T}}$. 按追随当前最优粒子的原理,粒子 i 将按下式改变速度和位置:

$$v_{id}^{n+1}=wv_{id}^{n}+c_1\mathrm{rand}()(p_{id}^{n}-x_{id}^{n})+c_2\mathrm{rand}()(p_{gd}^{n}-x_{gd}^{n}), \tag{1}$$

$$x_{id}^{n+1}=x_{id}^{n}+v_{id}^{n+1}. \tag{2}$$

其中:n 为当前的进化代数;c_1,c_2 为学习因子;rand() 为分布于(0,1)的随机数;w 为惯性权重. 研究表明,较大的 w 值有利于跳出局部极值点,较小的 w 值有利于算法的收敛和提高解的精度,惯性权重 w 起到平衡全局搜索和局部搜索能力的作用.

标准粒子群优化算法流程如下:

Step1:随机初始化粒子的位置和速度.

Step2:计算粒子的适应度值,将粒子的 p_i 设置为当前位置;p_g 设置为初始群体的最佳粒子的位置.

Step3:对所有粒子按式(1)和(2)更新位置和速度,并计算粒子的适应度值.

Step4:判断算法停止准则是否满足,若满足,则结束;否则,转 Step2.

3 动态分级的混合粒子群优化算法

3.1 算法原理

在现有的并行粒子群优化算法中,各个并行变量独立进行迭代搜索,并行变量之间缺乏信息交换适应度最优的并行解不能用来引导适应度差的解进行寻优,而且并行变量都采取一致的迭代方式,因而算法的整体优化效率不高,局部搜索能力不强. 根据优化算法中并行变量协同配合、信息交换的思想,本文提出了一种动态分级的混合粒子群优化算法(DHHPSO),其思路是:将并行变量分为 3 个级别,各个级别中的并行变量数目是:第 1 级为 n_1,第 2 级为 n_2,第 3 级为 n_3. 设算法并行变量总数目为 n,有 $n_1+n_2+n_3=n$. 在每次并行粒子群算法迭代后,n 组并行变量将有对应的适应度值,然后将其排序,将适应度值最优的 n_1 组并行解作为第 1 级,用于进行全局搜索. 搜索算法采用引入混沌机制的粒子群算法,这样能保持种群的多样性,提高算法的全局搜索能力. 第 2 级有 n_2 组用于全局结合局部的搜索,这里采用基本粒子群优化算法进行搜索,选适当的权重 w 平衡算法的全局搜索和局部搜索能力. 第 3 级有 n_3 组,主要用于局部搜索,采用单纯形法对适应度最优解进行局部搜索寻优,增强算法的局部搜索能力. 3 个级别中的并行变量经过一次寻优后,将各自的适应度值进行迭代,以此作为下次寻优的起点.

在算法中,搜索的初始阶段应偏重于全局搜索,考虑选用较多的并行变量进行全局搜索,因而此时 n_1 的值应较大些;在搜索中期,采取全局搜索和局部搜索并重的方式,则 n_1,n_2,n_3 的取值比较接近;而在搜索末期,主要考虑局部精确搜索,这时 n_3 的取值应大些,以提高局部搜索能力. 因此,每一级中并行变量的数目是动态调整的. 假设 $n=10$,在搜索初始阶段,取 $n_1=5,n_2=3,n_3=2$;在搜索中期,$n_1=3,n_2=4,n_3=3$;在搜索末期,$n_1=2,n_2=3,n_3=5$.

3.2 算法的步骤与流程

动态分级混合粒子群优化算法(DHHPSO)的流程如图 1 所示.

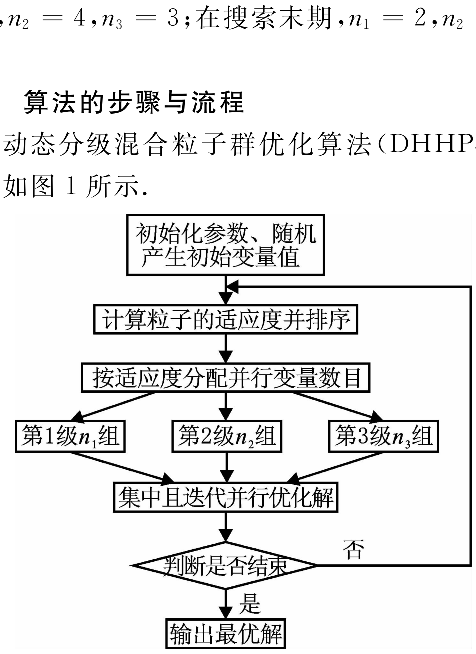

图 1 动态分级混合粒子群优化算法

具体步骤如下：

Step1：确定算法参数，随机产生初始变量.

Step2：计算所有粒子的适应度值并排序.

Step3：按适应度将并行变量分成 3 级：第 1 级为 n_1 组，第 2 级为 n_2 组和第 3 级为 n_3 组.

Step4：第 1 级用混沌粒子群混合优化算法进行全局搜索；第 2 级用基本粒子群算法进行全局结合局部搜索；第 3 级采用单纯形法进行局部搜索.

Step5：集中且迭代并行优化解.

Step6：判断算法是否结束. 若是，则输出最优解；否则，转 Step2.

3.3　算法的具体实现

本节主要说明分级部分算法的具体实现. 第 1 级主要是进行全局搜索，搜索算法采用引入混沌机制的粒子群算法，这样能提高算法的全局搜索能力. 第 2 级采用基本粒子群优化算法进行全局结合局部的搜索，算法中选取适当的惯性权重 w 平衡算法的全局搜索和局部搜索能力，具体的实现步骤与第 2 节中的算法步骤相同，这里不再赘述. 第 3 级主要用来进行局部搜索，采用单纯形法对适应度最优解进行局部搜索寻优，以增强算法的局部搜索能力，下面进行具体计算.

3.3.1　混沌粒子群优化算法实现全局搜索

混沌是一种非线性现象，具有遍历性、随机性和对初始条件的敏感性等特性，可在一定范围内按自身规律不重复地遍历所有状态，因此，可将其应用到优化算法中提高算法的全局搜索能力. 本文采用 Logistic 映射函数产生混沌变量. Logistic 映射是一个典型的混沌系统，即

$$Z_{n+1} = \mu Z_n(1 - Z_n). \tag{3}$$

当 $\mu = 4$ 时系统完全处于混沌状态.

设第 1 级有 n_1 组并行解，对每一次迭代后适应度最优的 n_1 个并行解进行混沌搜索. 混沌搜索算法的具体步骤如下：

Step1：确定适应度最优解. 每一次迭代后，根据各个并行解的适应度确定 n_1 组并行解，确定个体最优解.

Step2：n_1 组并行解中的所有粒子，通过式(3)产生新的混沌向量并线性变换到解空间，生成粒子新位置.

Step3：计算适应度值. 计算每个新粒子的适应度值，记录个体最优解及其适应度值，并与前面的个体最优解进行比较，更新最优解及适应度.

3.3.2　单纯形粒子群优化算法实现局部搜索

下面采用 N-M 单纯形法进行局部搜索. 单纯形法具有计算量小、优化速度快和局部搜索能力强等特点，因此将其应用于算法中以加强局部搜索能力. 下面对每一次迭代后的 n_3 组并行解进行一次单纯形计算，主要操作步骤如下：

Step1：确定适应度最优解. 在每一次迭代后，根据各个并行解的适应度确定 n_3 组并行解，确定个体最优解.

Step2：首先形成初始单纯形. 对于 n 维优化问题采用 n 个独立的单纯形，n_3 个并行解就是每个单纯形的 n_3 个初始顶点.

Step3：确定最大点、次大点和最小点. 进行反射、延伸、收缩和缩边操作.

4　数值仿真与分析

下面采用 3 个典型测试函数来评价所提出的动态分级的混合粒子群优化算法(DHHPSO)的性能，并与标准粒子群算法(SPSO)，嵌入混沌序列的混合粒子群算法(CPSO)[6] 和引入单纯形算子的混合粒子群算法(SMPSO)[10] 的测试结果进行比较.

函数 f_1(Spherical 函数)

$$f_1(x) = \sum_{i=1}^{D} x_i^2,\ -100 \leqslant x_i \leqslant 100,$$

是个单峰二次函数，它在 $x_i = 0$ 时达到最小值 0.

函数 f_2(Rosenbrock 函数)

$$f_2(x) = \sum_{i=1}^{D} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2],$$
$$-30 \leqslant x_i \leqslant 30,$$

是个很难极小化的病态二次函数，其极小点所在的山谷易找到，但要收敛到全局极小点则十分困难，在 $x_i = 1$ 时取得全局最小值 0.

函数 f_3(Rastrigin 函数)

$$f_3(x) = \sum_{i=1}^{D} [x_i^2 - 10\cos(2\pi x_i) + 10],$$
$$-5.12 \leqslant x_i \leqslant 5.12,$$

是个具有大量局部最优值的多峰函数，有很多正弦凸起的局部极小点，在 $x_i = 0$ 处取得全局最小值 0.

为了方便比较各个算法的性能，各算法的粒子群规模为 $n = 40$，最大进化代数 maxDT $= 500$. 为了进一步比较算法的性能和减少偶然性的影响，在相同迭代次数的条件下，各算法对每个函数的测试均运行 20 次，然后取平均值. DHHPSO，SPSO，CPSO 和 SMPSO 算法的寻优结果(包括找到的最优值和平均最优值)比较如表 1 所示. 对应的寻优曲线如图 2 ～ 图 4 所示.

从表 1 可知，对于 Spherical 函数，SPSO 算法和 SMPSO 算法的优化结果很不理想，不能找到最优值，且偏差较大. 相对于 SPSO 算法和 SMPSO 算法的优化结果，CPSO 算法的偏差较小，但也没有找到

表 1 4 种算法的优化结果比较

函数	算法	维数	找到的最优值	平均最优值
f_1	DHHPSO	30	0.00000349	0.00008846
	SPSO	30	0.21787501	0.35861203
	CPSO	30	0.00431938	0.01395034
	SMPSO	30	0.15360824	0.22305937
f_2	DHHPSO	30	25.7822046	31.2903582
	SPSO	30	55.2552793	80.3341561
	CPSO	30	34.8527381	41.7741200
	SMPSO	30	40.6502930	52.2263589
f_3	DHHPSO	30	0.03100514	0.10731142
	SPSO	30	46.2178291	50.3453679
	CPSO	30	4.102503368	7.36628970
	SMPSO	30	11.9246873	16.3158614

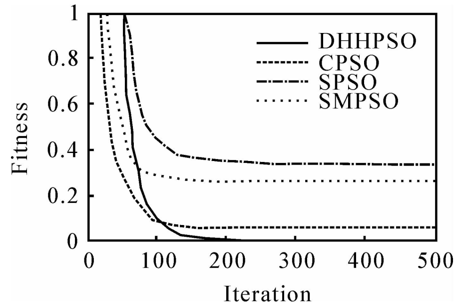

图 2 Spherical 函数的寻优曲线

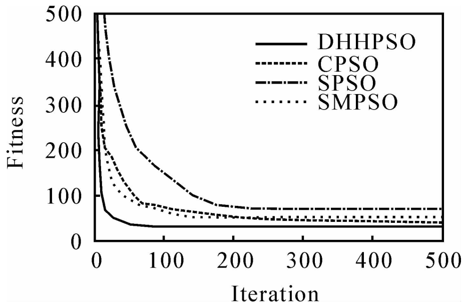

图 3 Rosenbrock 函数的寻优曲线

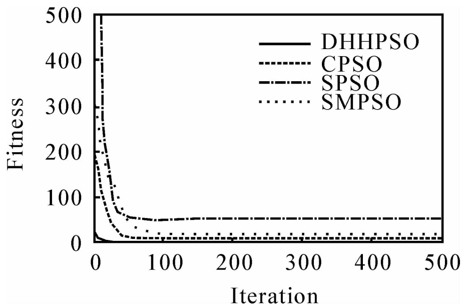

图 4 Rastrigin 函数的寻优曲线

最优值.这可能与算法本身不能对高维复杂问题进行寻优的局限性有关.而 DHHPSO 算法由于结合了全局寻优能力较强的混沌优化方法和局部搜索能力较强的单纯形法,大大增强了算法的全局搜索性能和收敛精度.从图 2 的寻优曲线可以看出,虽然 DHHPSO 算法的搜索性能较好,但与 SMPSO 算法和 CPSO 算法相比,其迭代前期收敛速度较慢,20 次实验均能较快地找到问题的最优值,且偏差较小. Rosenbrock 函数是个很难极小化的病态二次函数,要收敛到全局极小点十分困难,尤其是高维函数更是如此.表 1 和图 3 的优化结果也说明这点.对 Rastrigin 函数,从表 1 和图 4 可知,在相同演化代数的条件下,SPSO 算法的寻优能力和收敛结果最差,DHHPSO 算法的寻优性能最强,能收敛到最优值.

SPSO 算法是基于随机初始化的种群进行迭代求解,并使用适应度值来评价个体进行搜索,但不能保证一定找到全局最优值(有些复杂问题求最优值的近似值),尤其对于大规模的复杂问题,很容易陷入局部极值点;SMPSO 算法是在标准的粒子群算法的基础上加入单纯形算子,利用单纯形法较强的局部搜索能力进行搜索;CPSO 算法是在粒子群算法中嵌入混沌序列对粒子进行重新初始化,可确保算法有较强的全局搜索能力.DHHPSO 算法的优势是在于算法的混合,它利用两种或几种不同的算法进行结合,充分利用各种算法的优点来进行寻优. DHHPSO 算法在 PSO 算法的基础上,将粒子按适应度值的优劣分成 3 级,分别执行不同的搜索策略:采用混沌搜索的方法进行全局搜索;采用单纯形法进行局部搜索;同时应用标准粒子群算法平衡全局和局部搜索能力.这样,既能增强算法的全局搜索性能又能保证粒子群算法的优点.

5 结 论

仿真实验结果表明,所提出的动态分级的混合粒子群优化算法是一种有效的优化方法.通过动态自动调整粒子数目的方法确保了算法的全局搜索性能和局部搜索性能的动态平衡,它能够解决大量非线性、不可微和多峰值的复杂问题优化,并能获得较高的求解精度,在收敛性能上有显著提高.

参考文献(References)

[1] Kennedy J, Eberhart R C. Particle swarm optimization [C]. Proc of IEEE Int Conf on Neural Networks. Piscataway: IEEE Press, 1995: 1942-1948.

[2] Shi Y, Eberhart R C. Empirical study of particle swarm Optimization [C]. Proc of the 1999 Congress on Evolutionary Computation. Piscataway: IEEE Service Center, 1999: 1945-1950.

[3] Xie X F, Zhang W J, Yang Z L. A dissipative particle swam optimization[C]. Proc of the IEEE Int Conf on Evolutionary Computation. Honolulu: IEEE Inc, 2002: 1456-1461.

[4] Van den Bergh F, Engelbrecht A. A new locally convergent particle swarm optimization[C]. IEEE Int Conf on Systems, Man and Cybernetics. Hammamet, 2002: 96-101.

(下转第 1521 页)