差分演化算法在约束优化问题中的应用

廖 锋[1],高兴宝[2]
(1.南京师范大学 泰州学院 数学系,江苏 泰州 225300;
2.陕西师范大学 数学与信息科学学院,陕西 西安 710062)

摘 要:差分演化算法的变异机制没有充分利用种群的信息,导致变异是盲目的。受到粒子群算法信息共享机制的启发,文中提出了一种多群体差分演化算法,新算法将整个种群分成多个子种群,每个子种群通过借鉴本种群的内部经验与整个种群的外部经验对变异进行指导。一方面,由于变异操作借鉴了子种群的局部信息和整个种群的全局信息,提高了算法收敛的速度;另一方面,多个子群体增强了种群的多样性,提升了算法的全局搜索能力。数值实验表明新算法具有很强的稳定性和全局搜索能力,在相同计算复杂度情况下的全局搜索能力较原始差分演化算法有明显提升,可以有效求解约束优化问题。

关键词:差分演化算法;进化算法;粒子群算法;早熟
中图分类号:TP18 **文献标识码**:A **文章编号**:1673-629X(2010)05-0187-04

Application of Differential Evolution Algorithms on Constraint Optimization Problems

LIAO Feng[1], GAO Xing-bao[2]
(1. Department of Mathematics, Taizhou College, Nanjing Normal University, Taizhou 225300, China;
2. College of Mathematics and Information Science, Shaanxi Normal University, Xi'an 710062, China)

Abstract: The mutation mechanism of differential evolution algorithm doesn't apply the population information sufficiently, so the mutation operation is blind. Inspired by particle swarm optimization's information sharing mechanism, propose a multiple population differential evolution algorithm, which divides whole group into many sub-populations, thus the every sub-population's experiences from referring on inner message and outer message could instruct mutation operation. On the one hand, accelerating the convergence; on the other hand, increasing diversity of population. The numerical experiment indicated that the new algorithm has the features of stability and strong global exploration ability, which could solve constraint optimization problems efficiently.

Key words: differential evolution algorithm; evolution algorithm; particle swarm optimization; premature

0 引 言

差分演化(Differential Evolution, DE)算法[1]是由Storn.R和Price.K于1995年提出的一种基于实数编码的进化算法,其整体结构类似于遗传算法,采用变异算子提升个体间的差异性,采用杂交算子使得个体间的有用信息得到共享,采用最优保存策略保存种群的最优信息。由于其独特的杂交和变异方式使得DE具有强大的全局搜索能力,在1996年第一届国际IEEE进化计算竞赛上表现突出,被公认为是计算速度最快的进化算法。由于DE性能出众,在众多进化算法中脱颖而出并得到广泛应用[1,2]。

笔者考虑到DE对个体变异时没有挖掘出种群当前搜索到的有用信息,导致变异是盲目的、无指导的,而粒子群算法[3]在挖掘种群信息方面做的比较突出。受此启发提出了一种多群体差分演化 (Multiple Population Differential Evolution, MPDE)算法,将整个种群分成多个子种群,每个子种群按照自己的行为方式进行寻优,各个子群体在算法中的地位都是对等的,每个子群体的个体通过借鉴本种群的经验与整个种群的经验对变异进行指导。通过大量的数值实验表明,新算法可以有效解决约束优化问题。

1 差分演化(DE)算法

DE是一种随机并行直接搜索算法,以其简单实用

收稿日期:2009-10-04;**修回日期**:2010-01-05
基金项目:国家自然科学基金(60671063)
作者简介:廖 锋(1981-),男,硕士,助教,研究方向为进化计算,最优化理论与控制;高兴宝,博士,教授,博士生导师,研究方向为神经网络,最优化理论与控制。

性、较强稳健性以及强大的全局搜索能力使得DE算法在众多领域得到广泛应用，尤其在连续函数优化领域表现突出[1,2]。DE的整体结构类似于遗传算法，主要区别在于变异与杂交的方式上比较独特。设 $X_1, X_2, \cdots, X_N$ 表示当前种群，对 X_i 进行变异得到中间个体 v_i：

$$v_i = X_i + F \cdot (X_{r_1} - X_{r_2}) \tag{1}$$

其中 F 为缩放因子，一般在[0,2]范围内取值，r_1, r_2 为 $1,2,\cdots,N$ 中的两个随机数。在此基础上让 X_i 与中间个体 v_i 杂交得到实验个体 u_i：

$$u_{ij}(t) = \begin{cases} v_{ij}(t) & 若\ \mathrm{rand} \leqslant R\ 或\ j = r_3 \\ x_{ij}(t) & 否则 \end{cases} \tag{2}$$

其中 r_3 为 $1,2,\cdots,N$ 中的随机数。最后对 u_i 与 X_i 进行适应度比较，如果 u_i 的适应度比 X_i 的适应度高就让 u_i 替换 X_i。所以DE采用的是最优保存策略，种群搜索到的最优信息永远不会丢失，只要迭代次数足够大都能够以概率1收敛到全局最优解。

2 多种群差分演化(MPDE)算法

差分演化算法的全局搜索能力依赖于种群的多样性，如果算法使得各个体趋同，但是此时还不是全局最优点，则算法早熟收敛。从提升种群全局搜索能力角度考虑，将单种群进化算法改进为多个子种群协作的算法设计思想比较常见[4]。另外，DE对个体的变异操作(见式(1))是盲目、无指导的，而粒子群算法[2]在挖掘种群信息方面做得比较突出，通过借鉴种群的全局最优信息和个体的局部信息对个体的进化进行指导。

受此启发，文中提出了一种多种群差分演化算法(Multiple Popolation Differential Evolution, MPDE)：将整个种群分成多个子群体，某个子群体当前最优个体记为 L_{best}，表示子群体的最优信息；从所有子群体的最优个体中选出最优个体记为 G_{best}，表示群体的全局最优信息。在对个体进行变异操作时，将子种群的局部信息与整个种群的全局信息对变异操作进行指导，所以(1)式可改进为：

$$v_i = X_i + F_1 \cdot (G_{\text{best}} - X_{r_1}) + F_2 \cdot (L_{\text{best}} - X_{r_2}) \tag{3}$$

其中 F_1, F_2 为缩放因子，一般在[0,2]范围内取值，F_1 表示对全局最优信息的信任程度，称之为社会系数，衡量了个体间信息共享程度；F_2 表示对局部最优信息的信任程度，称之为认知系数，衡量了对自身经验的认知程度。

MPDE算法：

步骤1：初始化：随机初始化Pop_ mum个子群体，每个子群体规模为Pop_ Scale，个体 $X_i(0)$ 为Dimention维向量，表示搜索空间中的潜在解；随机初始化 G_{best} 以及 L_{best}。

步骤2：判断是否满足终止条件，如果满足则算法终止，否则转步骤3。

步骤3：对每个子群体进行如下操作：

步骤3.1：应用(3)式对 $X_i(t)$ 进行变异得到中间个体 $v_i(t)$。

步骤3.2：X_i 与 $v_i(t)$ 杂交得到中间个体

$$u_{ij}(t) = \begin{cases} v_{ij}(t)\ 若\ \mathrm{rand} \leqslant R\ 或\ j = r_3 \\ X_{ij}(t)\ 否则 \end{cases},$$

其中 r_3 是1与Dimention之间的随机整数；R 为杂交概率。

步骤3.3：根据适应度进行选择：

$$x_i(t+1) = \begin{cases} u_i(t), F(u_i(t)) > F(x_i(t)) \\ x_i(t), 否则 \end{cases}$$

步骤3.4：从群体 $x_i(t+1)$ 中确定最优个体作为该子群体的当前最优位置 L_{best}。

步骤4：对各个子群体的 L_{best} 进行适应度比较，更新 G_{best}，转步骤2。

该算法的特点是：各个子群体彼此的地位完全对等，在一个迭代周期(一般是次迭代)进行一次经验总结，评比出新的局部最优个体和全局最优个体，以便于对个体的变异进行指导。这样做的好处有两方面：一方面，由于变异操作(见式(3))借鉴了子种群的局部信息和整个种群的全局信息，提高了算法收敛的速度；另一方面，多群体增强了种群的多样性，提升了算法的全局搜索能力。

3 约束条件处理方案

考虑非线性规划：

$$\begin{cases} \min f(x) \\ \text{s.t. } g_i(x) \leqslant 0 (i = 1,2,\cdots,q) \\ h_j(x) = 0 (j = q+1, q+2, \cdots, m) \end{cases} \tag{4}$$

$F = \{x \mid g_i(x) \leqslant 0, h_j(x) = 0, i = 1,2,\cdots,q; j = q+1, q+2, \cdots, m\}$ 称为可行域，

$S = \{x \mid l_k \leqslant x_k \leqslant u_k, k = 1,2,\cdots,n\}$ 称为搜索空间，并且 $F \subset S$。定义违反约束度函数 $\text{violate}(x) = \sum_{i=1}^{q} \max\{g_i(x), 0\} + \sum_{j=q+1}^{m} | h_j(x) |$，显然对任意 $x \in R^n$ 都有 $\text{violate}(x) \geqslant 0$，$x \in F$ 当且仅当 $\text{violate}(x) = 0$。我们称 $\text{violate}(x)$ 为违反度函数，$\text{violate}(x)$ 越大说明点 x 违反约束程度越大即离可行域越远。

用差分演化算法求解非线性规划问题时不可避免地会产生不可行点，而处理好不可行点对求解约束优

化问题至关重要,文献[5]给出了一种处理不可行点的新方案:

$$\begin{cases} \text{case1: 当 violate}(x) \leqslant \varepsilon \text{ 且 violate}(y) \leqslant \varepsilon, \\ \quad \text{则目标函数值较小的为优。} \\ \text{case2: 当 violate}(x) \leqslant \varepsilon \text{ 且 violate}(y) > \varepsilon, \\ \quad \text{则 } x \text{ 为优。} \\ \text{case3: 当 violate}(x) > \varepsilon \text{ 且 violate}(y) > \varepsilon, \\ \quad \text{则违反程度较小的为优。} \end{cases} \quad (5)$$

方案(5)中有一个人为确定的违反约束度常数 ε,该方案将在违反约束度范围内的不可行点与可行点同等对待,这样做的好处是保留了处在可行域边界附近的点,这些点虽然不可行但含有丰富的有用信息,对搜索全局最优点很有帮助。

采用这种处理方案往往会使求得的全局最优点是不可行的但违反约束度 violate(x) 不超过,我们认为只要求出的全局最优解违反约束的程度很小,都是可以接受的。

4 数值模拟

4.1 测试函数

为测试MPDE算法性能,对文献[6]提供的13个经典的约束优化问题进行测试,这些测试函数是测试算法性能的标准函数,被广泛采用(参见文献[6~9])。

4.2 参数设置

参数设置:子群体数目 Pop_num=6,每个子群体规模 Pop_Scale=8,最大迭代次数为5000,社会系数 $F_1 = 0.9$,认知系数 $F_2 = 0.8$,杂交概率 $R = 0.5$,每个测试函数连续作20次独立试验。

4.3 试验结果

MPDE算法在Matlab7.0环境下编程实现,表1将MPDE算法的试验结果与最优值进行了横向比较。进一步,为了验证算法的有效性,将试验结果分别与其他5种算法的结果进行纵向比较,它们分别是HM[10],SR[6],ASCHEA[4],CHDE[7],SMES[8],比较结果见表2和表3。

表1 MPDE算法的测试结果

测试函数	最优值	违反约束度	最好结果	最差结果	方差	约束违反程度	成功概率
g01	−15	1.00E−04	−15.0005	−15.0005	0	1.00E−04	100%
g02	0.803619	1.00E−04	0.803554	0.763996	9.18E−05	0	505
g03	1	1.00E−03	1.0004	0.9897	9.89E−06	9.99E−04	95%
g04	−30665.54	1.00E−08	−30665.538	−30665.538	0	1.00E−08	100%
g05	5126.498	1.00E−04	5125.9518	5125.9518	0	1.00E−04	100%
g06	−6961.814	1.00E−07	−6961.814	−6961.814	0	1.00E−07	100%
g07	24.306	1.00E−05	24.31	24.384	4.28E−04	0	100%
g08	0.095825	1.00E−08	0.095825	0.095825	0	0	100%
g09	680.63	1.00E−06	680.6301	680.6304	6.66E−09	0	100%
g10	7049.25	1.00E−03	7044.874	7148.121	656.918	9.78E−04	30%
g11	0.75	1.00E−04	0.7494	0.75067	6.48E−08	5.00E−04	100%
g12	1	1.00E−06	1	1	1.00E−06	0	100%
g13	0.05395	1.00E−03	0.052683	0.43177	0.0342	1.00E−03	50%

表2 各算法最优结果比较

测试函数	最优值	HM	SR	ASCHEA	CHDE	SMES	MPDE
g01	−15	−14.7886	−15	−15	−15	−15	−15.0005
g02	0.803619	0.79953	0.803515	0.785	0.803619	0.803601	0.083554
g03	1	0.9997	1	1	1	1	1.0004
g04	−30665.54	−30664.5	−30665.539	−30665.5	−30668.539	−30665.539	−30665.538
g05	5126.498	—	5126.497	5126.5	5126.497	5126.599	5125.9518
g06	−6961.814	−6952.1	−6961.814	−6961.81	−6961.814	−6961.814	−6961.8139
g07	24.306	24.62	24.307	24.3327	24.306	24.327	24.31
g08	0.095825	0.095825	0.095825	0.095825	0.095825	0.095825	0.095825
g09	680.63	680.91	680.63	680.632	680.63	680.632	680.63
g10	7049.25	7147.9	7054.316	7056.113	7049.248	7051.903	7044.874
g11	0.75	0.75	0.75	0.75	0.749	0.75	0.7495
g12	1	0.9999	1	NA	1	1	1
g13	0.05395	NA	0.053957	NA	0.053866	0.053986	0.052683

表 3 各算法最差结果比较

测试函数	最优值	HM	SR	ASCHEA	CHDE	SMES	MPDE
g01	-15	-14.6154	-15	NA	-12.743	-15	-15.0005
g02	0.803619	0.79119	0.726288	NA	0.3021	0.7513	0.763995
g03	1	0.9978	1	NA	0.0296	1	0.9897
g04	-30665.54	-30645.9	-30665.539	NA	-29986.2	-30665.539	-30665.538
g05	5126.498	—	5142.472	NA	5304.167	5304.167	5125.9518
g06	-6961.814	-5473.9	-6350.262	NA	-6962.48	-6952.482	-6961.8139
g07	24.306	25.069	24.642	NA	24.843	24.843	24.384
g08	0.095825	0.02914	0.095825	NA	0.095825	0.095825	0.095825
g09	680.63	683.18	680.763	NA	680.719	680.719	680.63
g10	7049.25	9659.3	8835.635	NA	7638.366	7638.366	7048.121
g11	0.75	0.75	0.75	NA	0.75	0.75	0.7506
g12	1	0.99195	1	NA	1	1	1
g13	0.05395	NA	0.216915	NA	0.46829	0.46829	0.43177

4.4 试验备注与说明

表中“-”表示没有找到可行解，“NA”表示文献中没有该项数据。由于各个测试函数计算困难程度不一样，所以判定试验是否成功的准则也不尽相同，文中将计算结果与公布最优值的差的绝对值在某个阈值范围内就视为成功，否则视为失败，各测试函数的阈值设置见表 4。另外函数 g12 由于其特殊性，进行球坐标换元处理。

$$令\begin{cases} x_1 = R\cos\theta\cos\phi + p \\ x_2 = R\cos\theta\sin\phi + q \\ x_3 = R\sin\theta + r \end{cases}$$

经过这样的处理，原问题 g12 等价转化为：

$$g'12: \min f = [(R\cos\theta\cos\phi + p - 5)^2 + (R\cos\theta\sin\phi + q - 5)^2 + (R\sin\theta + r - 5)^2]/100 - 1$$

其中 $R \in [0,0.25]$，$\theta \in [0,2\pi]$，$\phi \in [0,\pi]$，$p,q,r \in [1,9]$。在换元前 g12 有 9^3 个约束条件，经过换元处理后约束条件变为 6 个，大大减轻问题的难度。

表 4 试验成功的判定标准

g01	g02	g03	g04	g05	g06	g07
0.0001	0.005	0.0005	0.0001	0.0001	0.0001	0.01
g08	g09	g10	g11	g12	g13	
0.0001	0.0005	1	0.0005	0.0001	0.005	

4.5 试验结果与分析

从表 1 看来，MPDE 找到 g04～g06，g08～g09，g12 的全局最优值与公布结果非常相近，甚至 g03，g04 的结果优于公布结果。g01，g02，g03，g11 的最优值与公布结果的误差也小于 5×10^{-4}。g07 由于问题的复杂性，最优值与公布结果的误差只能控制在 0～0.01 范围内。g10 对违反约束度 ε 的依赖性比较大，当 $\varepsilon = 0.001$ 时，兼顾了最优解的求解质量和算法的稳定性，但成功率只有 30%；虽然找的最优解与公布结果相差较远，但是 MPDE 找到的是比公布结果更好的结果，这个结果的不可行程度在误差允许范围内。另外从方差上看，除 g10 外都比较小，说明 MPDE 算法的比较稳定，并且全局搜索能力很强。

从表 2 来看，MPDE 算法在 g03，g11，g13 的计算结果不如其他算法，所以 MPDE 求解等式约束优化问题时在约束条件的处理方法上有待进一步改进。从 g05 的计算结果来看，不仅优于其他算法而且优于公布结果。虽然 g13 的计算结果优于公布结果和其他算法，但是最优解不是可行解，是在一定误差允许范围内的最优解。

5 结束语

DE 对个体变异时没有挖掘出种群当前搜索到的有用信息，导致变异是盲目的。为了充分利用种群的信息，文中提出一种多群体差分演化算法—MPDE，该算法与 DE 相比较，在计算复杂度大致相当的情况下，全局搜索性能得到有效提升。实验结果表明 MPDE 可以有效求解复杂的约束优化问题。

参考文献：

[1] Storn R, Price K. Differential Evolution - a Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces[R]. [s. l.]: International Computer Science Institute, 1995:22-25.

[2] 贺朝毅，王熙照. 基于改进 DE 算法的难约束优化问题的求解[J]. 计算机工程，2008，34(13)：193-195.

[3] Kennedy J, Eberhart R. Particle Swarm Optimization[C]// IEEE International Conference on Neural Networks, 1995. Piscataway, NJ: IEEE Service Center, 1995: 1942-1948.

[4] Hamida S B, Schoenauerm. ASCHEA: New Results Using Adaptive Segregation Constraint Handling[C]//Proceedings of the 2002 Congress on Evolutionary Computation. Piscataway: IEEE Service Center, 2002: 884-889.

[5] 廖 锋，高兴宝. 求解非线性规划问题的混合粒子群算法

（下转第 194 页）

同时根据粒度的分析,时间维最小分析单元是日,就可以把源数据时间格式转换成标准的日期格式(YY-MM-DD),按病人ID、科室名称、医生姓名、发票类别、日来汇总小计的金额。这样大大减少了记录条数,在对医院HIS的2007-2008年数据处理时,可以减少50%的记录。

3 物理模型的设计

数据仓库的物理模型的设计就是数据仓库逻辑模型在物理系统的实现。其中主要解决数据的结构类型、数据的索引策略、数据的存储策略及存储优化等问题。在进行物理模型的设计实现时,所考虑的因素有:性能价格比、I/O存储时间、空间利用率及维护的代价[8]。

以我院为例,每天产生的数据处方细节表就有几万条,细节数据量积累起来很快。物理设计时,把OLAP中使用频率高的、要求响应时间快的放在高速存储设备中。对于细节性数据、访问次数较少的放在低速、价格低廉的介质上。这样也可以提高高速存储设备的利用率,降低成本提高了性能价格比。

数据仓库数据量很大,建立索引会大大地提高访问效率。建立索引是要考虑空间成本的,并不是所有的表都需要索引,只需对访问比较频繁的表建立索引,同时根据时间需要选择索引的键。例如,在费用的关系表中,时间维度访问频率最高的,可以在事实表中建立TimeID的索引。

在设计存储策略中可以有许多方法提高I/O的效率,此处介绍这次数据仓库中用到的方法。

①增加冗余,将经常用到的字段添加到相应的表里,减少表的连接次数以及I/O的访问次数,提高访问速度。如,在开始设计病人维度时,病人类别是一个单独的关系表,其字段并没有放在病人维时,这样在分析费用中需要做两表的连接,降低了访问效率。

②归并表,即将需要同时访问的表顺序存放在同一个物理磁盘上,减少I/O访问的次数。

4 结束语

数据仓库的数据模型设计是数据仓库建设的核心,模型设计的好坏决定了数据仓库项目成功完成与否。文中以医院具体数据为背景讨论了概念模型、逻辑模型、物理模型的设计方法和主要内容。

医院建设数据仓库的数据源是HIS,数据范围在不断增加,数据字典时有变动,用户需求也随着不断变化,数据模型的设计本身就是个不断循环和修正的过程[9]。因此,医院数据仓库模型的设计也要遵循循环的开发方法,在用户看到实际结果后,又有可能提出更完善或新的需求,通过反馈使得数据模型不断的修正,最后形成适合医院的数据仓库模型。

参考文献:

[1] 段会龙,吕旭东.医疗信息系统发展现状及趋势[J].中国医疗器械信息,2004(10):1-6.

[2] 毛琦敏.数据仓库在医院应用的研究[J].医学研究生学报,2005,18(4):358-359.

[3] 胡黎玮,苏宁军.浅谈数据仓库成功实施的关键因素[J].科技情报开发与经济,2009,19(5):81-83.

[4] Inmon W H. 数据仓库[M].王海志,等译.北京:机械工业出版社,2006:27-58.

[5] Wisniewski M F, Kieszkowski P, Zagorski B M, et al. Development of a clinical data warehouse for hospital infection control [J]. Journal of the American Medical Informatics Association, 2003,10(5):455-461.

[6] 谷 岩,郭 庆.数据仓库系统中逻辑建模的方法研究[J].计算机系统应用,2005(8):42-46.

[7] Gyssens M, LakshmananL V S. A foundation for multi-dimensional databases[C]//Proceedings of the 23rd International Conference on Very Large DataBases. Athens, Greece: [s. n.],1997:107-114.

[8] 朱德利.SQL Server 2005数据挖掘与商业智能完全解决方案[M].北京:电子工业出版社,2007:123-126.

[9] Mallach E G. Decision Support and Data Warehouse Systems [M]. Beijing: Tsinghua University Press,2001:201-236.

(上接第190页)

[J].计算机工程与应用,2008,44(11):43-46.

[6] Thomas P R, Xin Y. Stochastic Ranking for Constrained Evolutionary Optimization[J]. IEEE Trans. on Evolutionary Computation, 2000,4(3):284-294.

[7] Mezura M E, Coello C A, Tun M I. Simple Feasibility rules and Differential Evolution for Constrained Optimization[C]//Proceedings of the 3rd Mexican International Conference on Artificial Intelligence, LNCS 2972. Heidelberg: Springer-Verlag,2004:707-716.

[8] Mezura M E, Carlos A, Coello C A. Simple Evolution Strategy to Solve Constrained Optimization Problems[J]. IEEE Trans. on Evolutionary Compution,2005,9(1):1-17.

[9] 熊 敏,刘玉树.基于协同进化遗传算法的地域选取算法[J].计算机技术与发展,2006,16(6):174-176.

[10] Koziel S, Michalewicz Z. Evolutionary Algorithms, Homomorphous Mappings, and Constrained Parameter Optimization [J]. Evolutionary Compution,1999,7(1):19-44.