具有参数自适应机制的改进离散差分进化算法

王丛佼 王锡淮 肖建梅
(上海海事大学电气自动化系 上海 201306)

摘 要 在研究和分析离散差分进化算法的基础上,提出了一种具有参数自适应机制的改进离散差分进化算法(PA-DDE)。该算法首先对连续域进化过程中的参数进行自适应调整,以平衡全局搜索与局部搜索,协调种群多样性和收敛速度间的矛盾,其次根据对应离散域上成功进化的个体的离散编码反馈信息引导算法协同进化。通过对背包问题进行的实验表明,该算法具有良好的收敛效率和稳定性。

关键词 离散差分进化,参数控制,离散编码,协同进化,多维背包问题
中图法分类号 TP18 **文献标识码** A

Improved Discrete Differential Evolution with Parameter Adaptive Mechanism

WANG Cong-jiao WANG Xi-huai XIAO Jian-mei
(Department of Electrical Engineering and Automation, Shanghai Maritime University, Shanghai 201306, China)

Abstract An improved discrete differential evolution algorithm (PA-DDE) with mechanism of parameter adaptive was proposed, based on the research and analysis of discrete differential evolution algorithm. Firstly, the parameters of continuous domains are adaptively adjusted in the process of evolution to balance the global search and local search, and also to coordinate the contradiction of population diversity and convergence speed. Secondly, the co-evolution processing is guided by the feedback information of discrete encoding of the successful evolutionary individuals on the corresponding discrete domains. Simulation results on the knapsack problem show that the proposed algorithm has good convergence efficiency and stability.

Keywords Discrete differential evolution, Parameter control, Discrete encoding, Co-evolution, Multidimensional knapsack problem

1 引言

差分进化(Differential Evolution, DE)算法是由 Storn 和 Price 于 1995 年提出的一种基于群体差异的全局优化算法[1]。该算法具有结构简单、容易实现、鲁棒性强等特点[2,3],其基本思想是:基于种群个体的差异重组得到一个由父子混合个体构成的中间种群;再由中间种群与原始种群的个体通过优胜劣汰的原则形成新一代种群。DE 算法在提出之初主要采用实数编码,因此一般用于求解实数优化,并且已在解决连续域最优化问题上得到了广泛和成功的应用[4-6]。然而其在求解离散型最优化问题方面的研究成果较少,因此如何将 DE 算法更好地应用于离散领域是一个值得关注的研究方向。

近年来,为了利用 DE 算法处理离散优化问题,Pampará 等提出一种仍采用实数编码的二进制 DE 算法[7],它通过一个三角函数将实数个体转换为二进制串;Gong 和 Tuson 在文献[8]中提出一种基于正则分析的二进制 DE 算法,即通过正则分析对传统 DE 算法的变异和交叉算子进行改动,并通过组合优化问题验证了算法的可行性;Moraglio 和 Togelius 在研究变异算子几何特性的基础上,通过海明距离提出了二进制 DE 算法[9];Greenwood 则在文献[10]中提出了一种简单的离散转换机制,并成功用于图论问题的求解中。

但上述算法将 DE 引入离散域时要么是采取一种简单映射关系,要么是对连续数值进行固定区段的划分或完全随机划分,未能考虑实际问题的特性,从而导致算法可能出现后期收敛速度变慢或由于种群多样性的损失而陷入局部最优等问题。本文将自适应和协同进化的思想引入到离散 DE 算法中,形成了具有参数自适应的离散差分进化算法(Improved Discrete Differential Evolution with Parameter Adaptive Mechanism, PA-DDE)。DE 算法对参数设置非常敏感,不同的实际问题需要设置不同的参数,不合理的参数选取会降低算法的收敛效率并可能导致其过早收敛,影响算法稳定性。本文提出的 PA-DDE 算法通过参数自适应机制提高 DE 算法的收敛性能,同时基于协同进化的思想,通过离散断点因子实现实数空间与离散空间的联系,并根据离散编码的反馈结果来引导 DE 算法的有效进化方向。最后通过多维背包问题的仿真实验验证了该算法的有效性。

2 标准 DE 算法

DE 算法和其他进化算法一样,是一种模拟生物进化的

到稿日期:2013-03-07 返修日期:2013-07-24 本文受上海市教委科研创新重点项目(12ZZ158),上海市教委重点学科建设项目(J50602)资助。
王丛佼(1984—),女,博士生,主要研究方向为智能控制、智能信息处理等,E-mail:qinxiaojiayi@hotmail.com;王锡淮(1961—),男,教授,博士生导师,主要研究方向为智能控制、智能信息处理、智能图像处理等;肖建梅(1962—),女,教授,主要研究方向为智能控制、智能信息处理等。

随机模型，通过反复地迭代，只有适应环境的个体生存下来。DE算法的原理与遗传算法相似，主要包括变异、交叉和选择3个基本进化步骤，而其独特的差分变异操作和一对一的竞争生存策略不仅保留了基于种群的全局搜索策略，更降低了遗传操作的复杂性。

对于最小化优化问题：

$\min f(x)$

s. t. $x\in X$

其中，$x\in R^n$ 是目标变量，$f(x)$ 为适应度函数，$X\subset R^n$ 为约束集或可行域。

DE算法的基本流程如下：

1)种群初始化。设定种群规模为 NP，每个个体自变量有 D 维。种群中第 i 个个体 x_{ij} $(i=1,\cdots,NP;j=1,\cdots,D)$ 为一实数在其自变量范围 $[l_j,h_j]$ 内随机均匀初始化。

2)变异操作。DE算法利用种群中不同个体间的差分向量对个体进行扰动，产生变异个体 v_i^{G+1}。变异策略可以有多种，一般比较常用的有：

DE/rand/1：

$$v_i^{G+1}=x_{r1}^G+F(x_{r2}^G-x_{r3}^G) \tag{1}$$

DE/best/1：

$$v_i^{G+1}=x_{best}^G+F(x_{r2}^G-x_{r3}^G) \tag{2}$$

式中，$r_1,r_2,r_3\in[1,2,\cdots,NP]$，为互不相同的整数；$G$ 为当前迭代次数；x_{best}^G 为第 G 次迭代的当前最优解；F 为缩放因子，$F\in(0,2)$。

3)交叉操作。DE算法的交叉算子分为二项式交叉和指数交叉。常用的二项式交叉算子表示为：

$$u_{ij}^{G+1}=\begin{cases}v_{ij}^{G+1}, & rand(j)\leqslant CR \text{ or } j=randn(i)\\ x_{ij}^G, & rand(j)>CR \text{ and } j\neq randn(i)\end{cases} \tag{3}$$

式中，$rand(j)\in[0,1]$ 为均匀分布的随机数；$randn(i)\in[1,2,\cdots,D]$ 为随机选择的维数索引，其保证 u_i^{G+1} 中至少有一维由变异向量贡献；交叉概率因子 $CR\in[0,1]$。

4)选择操作。DE算法采用"贪婪"的选择策略，经变异交叉操作后生成的试验个体与目标个体进行竞争，适应度值更优的个体保存到下一代种群中。其选择算子可描述为：

$$x_i^{G+1}=\begin{cases}u_i^{G+1}, & f(u_i^{G+1})<f(x_i^{G+1})\\ x_i^G, & f(u_i^{G+1})\geqslant f(x_i^{G+1})\end{cases} \tag{4}$$

5) 终止条件。如果迭代次数 G 达到最大迭代次数 $G_{\max}$ 或者最优个体适应度值的精度达到要求，则算法停止搜索；否则，种群继续执行变异、交叉和选择操作。

3 PA-DDE算法

传统DE算法无法直接对离散域上的最优化问题进行求解，利用该算法求解离散型最优化问题的关键在于如何将实数域的DE算法与离散域上待求解的问题联系起来。本文受拓扑空间思想启发，提出了一种协同进化空间的构建，该方法将实数域上的算子通过自适应的离散断点转换为二进制编码，并且利用成功进化个体的离散编码的适应度值来引导算法的进化，从而通过拓扑离散空间和对应实数空间上的协同进化来达到求解离散域最优化问题的目的。

3.1 协同进化空间的构建

协同进化空间的构造机制为：首先，将离散域上最优化问题的解扩展为 NP 个候选解向量，使之与实数域中的 NP 个 D 维向量一一对应，初始的离散域候选解向量由实数域产生。其次，实数域上 D 维向量值的改变通过离散断点算子反馈至离散领域，产生新一代的离散域候选解向量；计算离散域候选解向量的适应度值，反馈至实数域引导算法协同进化的方向，重复这一过程，从而实现离散域上二进制编码解向量的进化。

定义1 假设离散空间(Discrete Space，DS)上的最优化问题的解为 D 维向量，则设定协同进化的实数空间(Real-number Space，RS)中种群规模为 NP，每个个体自变量有 D 维。种群中第 i 个个体为 X_i，组成 X_i 的每一维表示为 x_{ij} $(i=1,\cdots,NP;j=1,\cdots,D)$，$X_i$ 在其自变量范围 $[l_j,h_j]$ 内随机均匀初始化。相应的协同进化离散空间(Co-evolution Discrete Space，CDS)中种群规模也为 NP，每个个体自变量有 D 维。种群中第 i 个个体为 $Y_i=\{y_{i1},y_{i2},\cdots,y_{ij}.,y_{iD}\}$ $(i=1,\cdots,NP;j=1,\cdots,D,y_{ij}\in\{0,1\})$。其中：

$$y_{ij}=\begin{cases}1, & x_{ij}\geqslant Q_j\\ 0, & x_{ij}\leqslant Q_j\end{cases},i=1,2,\cdots,NP,j=1,2,\cdots,D \tag{5}$$

式中，Q_j 为离散断点算子，$Q_j\in[l_j,h_j]$。通过实数空间的解 X_i 对应拓扑离散空间的解 Y_i 来实现协同进化。

证明：由离散空间的性质可知，拓扑空间是离散的，当且仅当它的单元素集合是开集，换言之，当且仅当它不包含任何会聚点。CDS中的元素可表示为集合 $CDS=\{Y_1,Y_2,\cdots,Y_{NP}\}$，其中 $Y_i=(y_{i1},y_{i2},\cdots,y_{iD})$。CDS的单元素集合均为开集，综上CDS为对应的RS的离散拓扑空间。

定义2 设 $X_{i,G}$ 表示DE算法第 G 代RS上的第 i 个个体，则 $Y_{i,G}$ 表示DE算法第 G 代CDS上的第 i 个对应个体，$Y_{i,G}=(y_{i1,G},y_{i2,G},\cdots,y_{iD,G})$，其中 $y_{ij,G}$ 可由式(5)得出。

定义3 设离散空间上最优化问题 $f(y)=f(y_1,y_2,\cdots,y_D)$，$y_i\in\{0,1\}$，$G$ 和 $G_{\max}$ 分别表示当前进化代数和最大进化代数，将算法中第 G 代CDS上的个体按照函数适应度值排序，选取适应度值最好的个体 CX_G^{best}。

定义4 设 $V_{i,G+1}$ 是算法第 $G+1$ 代RS上中间种群中的第 i 个变异个体，F 为缩放因子，CR 为交叉概率因子，X_G^{best} 为第 G 代种群中RS上 CX_G^{best} 的协同进化个体，$X_{r1,G}$，$X_{r2,G}$ 为第 G 代种群中RS上的两个随机个体。

$$X_G^{best}=\{x_{1,G}^{best},x_{2,G}^{best},\cdots,x_{D,G}^{best}\} \tag{6}$$

则实数空间上的差分算子(Real-number Space Differential Operator，RSDO)定义为：

$$V_{i,G+1}=(v_{i1,G+1},v_{i2,G+1},\cdots,v_{iD,G+1}) \tag{7}$$

其中：

$$v_{ij,G+1}=\begin{cases}x_{j,G}^{best}+\\ F_{i,G}(x_{r1j,G}-x_{r2j,G}), & rand(j)\leqslant CR_{i,G} \text{ or } j=randn(i)\\ x_{ij,G}, & rand(j)>CR_{i,G} \text{ and } j\neq randn(i)\end{cases} \tag{8}$$

$i=1,2,\cdots,NP;j=1,2,\cdots,D$；$F_{i,G}$ 和 $CR_{i,G}$ 由本文3.2节得到。

定义5 协同进化离散空间上的差分算子(Co-evolution Discrete Space Differential Operator，CDSDO)定义为：

$$E_{i,G+1}=(e_{i1,G+1},e_{i2,G+1},\cdots,e_{iD,G+1}) \tag{9}$$

其中：

$$e_{ij,G+1}=\begin{cases}1, & v_{ij,G}\geqslant Q_{j,G}\\ 0, & v_{ij,G}<Q_{j,G}\end{cases} \tag{10}$$

定义6 设 $X_{i,G+1}$ 表示算法中第 $G+1$ 代实数空间上的

个体，$E_{i,G+1}$表示算法中第$G+1$代协同进化离散空间上的差分算子，$X_{i,G}$表示算法第G代实数空间上的个体，$Y_{i,G}$表示第G代协同进化离散空间上的个体。则实数空间选择算子(Real-number Space Select Operator，RSSO)可表示为：

$$X_{i,G+1}=\begin{cases}V_{i,G+1}, & f(E_{i,G+1})<f(Y_{i,G})\\ X_{i,G}, & f(E_{i,G+1})\geqslant f(Y_{i,G})\end{cases} \tag{11}$$

定义 7 对应的协同进化离散空间选择算子(Co-evolution Discrete Space Select Operator，CDSSO)可表示为：

$$Y_{i,G+1}=\begin{cases}E_{i,G+1}, & f(E_{i,G+1})<f(Y_{i,G})\\ Y_{i,G}, & f(E_{i,G+1})\geqslant f(Y_{i,G})\end{cases} \tag{12}$$

3.2 参数自适应策略

传统 DE 算法对参数的选择非常敏感，参数设置会影响算法的收敛性能和寻优效率[11-12]。其中缩放因子 F 对收敛速度影响较大，而交叉概率因子 CR 往往与所求问题的性质相关。为进一步提高 DE 算法的寻优性能，本文提出的 PA-DDE 算法中对 CR 和 F 进行动态自适应的调整，其基本思想是：采用正态分布生成 CR，通过新的调整规则，利用使试验个体成功进入下一代的 CR 引导产生新的 CR 值；对于 F 则选取柯西算子作为其生成算子，同时应设置较大的进化初期 F 值，以保持种群多样性，增大搜索到全局最优解的概率，而在进化后期 F 值应趋小，使个体向最优解快速靠拢，促进种群收敛，这可通过调整柯西分布的位置参数来实现。

3.2.1 自适应交叉概率因子

正态分布既有其规律性又带有随机性，本文利用正态分布随机数产生 CR，并根据成功进化个体的适应度变化值和相应的 CR 贡献度来更新正态分布。

算法中每个个体的交叉概率因子 $CR_{i,G}$表示为：

$$CR_{i,G}=N(CR_{PA},\sigma_{PA}^2) \tag{13}$$

式中，$N(CR_{PA},\sigma_{PA}^2)$表示均值为 CR_{PA}、方差为 σ_{PA}^2 的正态分布随机函数。当 $CR_{i,G}$的值超出[0,1]范围时，则采用截断的方式使其在[0,1]区间。在每一代进化过程中，试验个体成功进入下一代时所对应的 CR 值被保存在一个数组 M_{CR}，同时在数组 $M_{\Delta f}$保存成功进化个体的适应值修正度，其定义公式如下：

$$\Delta f(k)=\frac{f(k)-f_{new}(k)}{f(k)} \tag{14}$$

式中，$f(k)$和 $f_{new}(k)$分别表示个体进化前后的适应度值。则每一个成功进化个体的交叉概率因子的贡献度可表示为：

$$\omega_k=\frac{\Delta f(k)}{\sum_{k=1}^{|M_{\Delta f}|}\Delta f(k)} \tag{15}$$

正态分布的方差 σ_{PA}^2对算法性能影响不大，设为 0.01 不变，而均值 CR_{PA}初始值设为 0.5，之后按式(16)更新：

$$CR_{PA}=\sum_{k=1}^{|M_{CR}|}\omega_k\times M_{CR}(k) \tag{16}$$

其中$|M_{\Delta f}|=|M_{CR}|$，即为成功进入下一代的试验个体的个数。

3.2.2 自适应缩放因子

与正态分布相比，柯西分布其两翼分布更广的特性不仅可以提供多种步长选择，而且能增加搜索步长的广度，从而扩大寻优范围，帮助算法摆脱局部极值点的干扰。本文算法中第 G 代个体的缩放因子 $F_{i,G}$按如下公式独立产生：

$$F_{i,G}=C(F_{PA},r) \tag{17}$$

式中，$C(F_{PA},r)$表示按照位置参数 F_{PA} 和缩放参数 r 产生的柯西分布随机数。由于较小的 F 值容易使算法局部收敛，而较大的 F 值会降低收敛速度，甚至导致算法难以收敛，因此将 F 控制在截断区间$[F_{\min},F_{\max}]$($F_{\min}$一般取 0.2，$F_{\max}$一般取 0.8)。本文设置 $r=0.05$，F_{PA}的初始值取 0.7，之后每一代按式(18)根据进化代数动态更新：

$$F_{PA}=F_{\max}-(F_{\max}-F_{\min})\times\left(\frac{G}{G_{\max}}\right)^{\frac{1}{2}} \tag{18}$$

3.3 离散断点算子 Q_j 的自适应策略

协同进化的效果关键在于离散断点算子 $Q_{j,G}$的选取。不同于一般离散 DE 算法中固定不变的离散算子，本文算法根据成功进化的个体自适应改变 $Q_{j,G}$的设定值。

定义 8 设初始值 $Q_{j,0}=\frac{l_j+h_j}{2}$，第 G 代到第 $G+1$ 代的进化中，有 k 个协同进化离散空间上的差分算子(CDSDO)实现了成功进化，标记这 k 个成功进化的算子为：

$$E_{i,G+1}^{Sucess}=(e_{i1,G+1}^{Sucess},e_{i2,G+1}^{Sucess},\cdots,e_{iD,G+1}^{Sucess}),i=1,2,\cdots,k$$

相应地，有 $NP-k$ 个进化失败的算子，标记为：

$$E_{i,G+1}^{Fail}=(e_{i1,G+1}^{Fail},e_{i2,G+1}^{Fail},\cdots,e_{iD,G+1}^{Fail}),i=1,2,\cdots,NP-k)$$

则定义算法中第 $G+1$ 代的离散断点因子 $Q_{j,G+1}$为：

$$Q_{j,G+1}=Q_{j,G}-\sum_{i=1}^{k}\frac{h_j-l_j}{4\times NP}\times e_{ij,G+1}^{Sucess}+\sum_{i=1}^{NP-k}\frac{h_j-l_j}{4\times NP}\times e_{ij,G+1}^{Fail} \tag{19}$$

式(19)根据每一维在协同进化离散空间上的表现动态地调整离散断点因子 $Q_{j,G}$，使得表现更好的维被选为“1”的概率增大，相应地，表现不好的维被选为“0”的概率增大。

3.4 贪婪策略

对于超出实际约束条件的解，通常采取贪婪策略进行修正：对每一个新生成的离散空间的解向量，判断是否超出约束条件。如果超出，则令解向量的离散编码为“1”的维中价值/体积比最低的维等于 0；同时随机选取该解向量的离散编码中为“0”的维在实数空间的值，对实数空间中对应个体(或算子)的相应维进行替换。重复上述步骤，直到满足约束条件。

3.5 PA-DDE 算法步骤

步骤 1 设定参数初始值 CR_{PA}和 F_{PA}。初始化种群，$X_{i,0}$在定义域内随机初始化，根据 $Q_{j,0}=(l_j+h_j)/2$ 计算 $Y_{i,0}$。

步骤 2 根据式(13)、式(17)计算每个个体的交叉概率因子 $CR_{i,G}$和缩放因子 $F_{i,G}$；

步骤 3 根据式(7)－式(10)计算实数空间和协同进化离散空间的差分算子；

步骤 4 采用 3.4 节的贪婪策略处理结果；

步骤 5 根据式(11)、式(12)计算实数空间和协同进化离散空间的选择算子；

步骤 6 记录成功进化的试验个体，根据式(19)重新计算相应的离散断点因子 $Q_{j,G+1}$；

步骤 7 根据式(14)－式(16)更新 CR_{PA}，采用式(18)更新 F_{PA}；

步骤 8 判断算法是否达到终止条件，如果算法满足终止条件，则输出结果；否则，转向步骤 2。

4 仿真实验

为了验证算法的性能，通过多维背包问题测试实例将 PA-DDE 算法与遗传算法(GA)、离散粒子群算法(BPSO)[13]

和基于二元编码的差分进化算法(BDE)[8]进行了比较。多维背包问题(Multidimensional Knapsack Problem,MKP)是经典的NP难组合优化问题[14],其典型应用包括资本预算、货物装载和存储分配等。多维背包问题的一般形式可表述为:设物体具有价值和体积属性,每个背包具有容积属性,从n个物体里挑选合适的物体装入m个背包中,使得在不超出背包容积的条件下价值达到最大。

所有实验均在CPU为Intel Core™ i3 2.2GHz、内存为4GB的PC机上运行,并采用VC++6.0编程实现。各个算法的参数设置如下:GA算法选取单点交叉和单点变异,其交叉概率和变异概率分别为0.5和0.1;BPSO算法选取$c_1=c_2=2$,$\omega=1$;BDE算法变异因子$F=0.5$,交叉因子$CR=0.8$。所有算法种群规模均为100,总迭代次数5000代。从经典的多维背包问题测试集中选择6个实例:WEING1,WEING5,WIENG6,WIENG8,WEISH13和WEISH14进行验证,算法运行100次的统计结果如表1所列。

从表1可以看出,在处理简单实例WEING1和WEING6时,GA算法具有最少的平均迭代次数,PA-DDE算法其次,BPSO算法的平均迭代次数最多。而在处理其他复杂实例时,GA算法与BPSO算法均未达到理想的效果,BDE算法在大部分实例上虽取得最优解,但成功率远远低于PA-DDE算法,而平均迭代次数多于PA-DDE算法。由此可见,随着实例复杂度的增大,GA算法和BPSO算法的可满足样例数迅速减小并趋于0,但PA-DDE算法仍能保持较高的可满足样例数,并且其全局收敛速度比其他算法更快。

表1 4种算法平均迭代次数和成功率比较

实例	平均迭代代数				成功率			
	GA	BPSO	BDE	ACDDE	GA	BPSO	BDE	ACDDE
WEING1	104.6	290.3	137.8	125.4	100%	100%	100%	100%
WEING5	4793.2	4993.8	4189.1	3284.1	11%	2%	72%	94%
WEING6	61.5	129.1	75.4	69.7	100%	100%	100%	100%
WEING8	>5000	>5000	4890.7	4378.9	0%	0%	53%	76%
WEISH13	4874.1	>5000	4390.8	3765.7	7%	0%	61%	87%
WEISH14	>5000	>5000	4776.9	3351.2	0%	0%	56%	92%

由于4种算法在实例WEING1和WEING6中全部找到最优解,抽取4种算法50代时在这两个实例上的数据进行比较,比较结果如表2所列。

表2 4种算法进化50代时在实例WEING1和WEING6的比较

问题实例	已知最优解	GA			BPSO			BDE			PA-DDE		
		最好解	平均解	标准差	最好解	平均解	标准差	最好解	平均解	标准差	最好解	平均解	标准差
WEING1	141278	141278	141012.36	77.12	141278	140789.74	109.12	141278	141187.14	39.65	141278	141235.43	27.09
WEING6	130623	130623	129978.92	104.56	130623	128573.29	216.81	130623	130409.21	199.95	130623	130478.47	132.53

由表2可知,尽管表1中4种算法在实例WEING1和WEING6上迭代5000次,最终都取得了最优解,但参考4种算法在迭代50代时(略小于找到全局最优解的平均迭代次数)的表现,从平均解和标准差两方面来比较,PA-DDE算法均优于GA、BPSO和BDE算法,这说明PA-DDE算法在趋向全局最优解的迭代过程中不仅具有更快的寻优速度,同时还具有更好的算法稳定性。

结束语 本文提出了一种带参数自适应的协同进化离散差分进化算法(PA-DDE)。PA-DDE算法对参数进行动态调整,缩放因子和交叉概率因子根据不同的优化阶段而自适应变化,提高了DE算法的收敛性能和稳定性;通过构造协同进化的离散空间,同时结合自适应离散断点算子,实现了实数空间与离散空间的联系,并使算法具有更好的协同进化效果。求解多维背包问题的实验结果验证了PA-DDE算法在处理离散域最优化问题时的良好性能。

参考文献

[1] Price K, Storn R. Differential evolution-A practical approach to global optimization [M]. Berlin, Germany: Springer-Verlag, 2006:133-152

[2] 袁俊刚,孙治国,曲广吉.差异演化算法的数值模拟研究[J].系统仿真学报,2007,19(20):4646-4648

[3] Mezura-Montes E, Miranda-Varela M E. Differential evolution in constrained numerical optimization: An empirical study [J]. Information Sciences, 2010, 10(22): 4223-4262

[4] Dick G, Whigham P. Spatially-structured sharing technique for multimodal problems [J]. Journal of Computer Science and Technology, 2008, 23: 64-76

[5] 刘荣辉,郑建国.分区交叉差分进化算法及其约束优化[J].计算机科学,2012,39(2):283-287

[6] Gong W, Cai Z. An improved multiobjective differential evolution based on pareto-adaptive ∈-dominance and orthogonal design [J]. European Journal of Operational Research, 2009, 198(2): 576-601

[7] Pampará G, Engelbrecht A P. Binary differential evolution[C]// 2006 IEEE Congress on Evolutionary Computation. 2006: 1873-1879

[8] Gong T, Tuson A L. Differential evolution for binary encoding [C]// Soft Computing in Industrial Applications, ASC39. 2007: 251-262

[9] Moraglio A, Togelius J. Geometric Differential Evolution[G]// Proceedings of Genetic Evolutionary Computation Conference. 2009: 1705-1712

[10] Greenwood G W. Using differential evolution for a subclass of graph theory problem[J]. IEEE Transactions on Evolutionary Computation, 2009, 13(5): 1190-1192

[11] Qin A K, Huang V L, Nuganthan S P. Differential evolution algorithm with strategy adaptation for global numerical optimization[J]. IEEE Transaction on Evolutionary Computation, 2009, 13(2): 398-417

[12] Brest J, Greiner S, Boskovie B. Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems [J]. IEEE Transactions on Evolutionary Computation, 2006, 10(6): 646-657

[13] 沈显君,王伟武,郑波尽,等.基于改进微粒群优化算法的0-1背包问题求解[J].计算机工程,2006,32(18):23-38

[14] Pisinger D. Where are the hard knapsack problem [J]. Computer&Operations Research, 2005, 32(9): 2271-2284