软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2015,26(7):1557−1573 [doi: 10.13328/j.cnki.jos.004613]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

类搜索算法*

陈 皓, 潘晓英

(西安邮电大学 计算机学院,陕西 西安 710121)

通讯作者: 陈皓, E-mail: chenhao@xupt.edu.cn

摘 要: 提出利用类结构驱动的群体进化计算方法——类搜索算法(CSA).CSA 在个体间构造簇类形态的虚拟连接关系,并通过对类组织的结构和类搜索过程进行动态调节来优化模拟进化系统的计算状态,提高群体的搜索效率.介绍了 CSA 的基本模型,并基于 CSA 融合进化算子与差分计算机制设计出数值优化算法 CSA/DE.对多个典型高纬函数和复杂混合函数的仿真实验结果说明,CSA/DE 是一种对高纬连续问题高效、稳定的搜索优化方法.该工作一方面验证了 CSA 的可行性和有效性;另一方面则显示:基于类搜索模型可有效融合异构且具有不同计算特性的搜索机制,形成对待求解问题更具针对性且协调性更佳的搜索计算方法.这为高性能优化算法的设计提供了一条新的途径.

关键词: 进化算法;类进化优化模型;类搜索机制;数值优化

中图法分类号: TP301

中文引用格式: 陈皓,潘晓英.类搜索算法.软件学报,2015,26(7):1557−1573. http://www.jos.org.cn/1000-9825/4613.htm

英文引用格式: Chen H, Pan XY. Clustering search algorithm. Ruan Jian Xue Bao/Journal of Software, 2015,26(7):1557−1573 (in Chinese). http://www.jos.org.cn/1000-9825/4613.htm

Clustering Search Algorithm

CHEN Hao, PAN Xiao-Ying

(School of Computer Science and Technology, Xi'an University of Post & Telecommunication, Xi'an 710121, China)

Abstract: A novel evolutionary optimization method, clustering searching algorithm (CSA), is presented. In CSA, a virtual cluster group is constructed among individuals in order to adjust the operation state of simulated evolutionary system dynamically and improve the searching efficiency of population. After introducing the basic model of CSA, this paper presents CSA/DE, a new CSA blending the evolutionary search operators with the differential computing mechanism for solving numerical optimization problem. In simulations, 6 classical multidimensional functions and 6 challenging composition functions are selected to test the performance of CSA/DE. The experimental results show CSA/DE is an efficient and reliable search optimization algorithm for multidimensional continuous problem. The work of this paper verifies the feasibility and validity of CSA. Meanwhile, this research demonstrates, based on CSA, multiple heterogeneous searching mechanisms can be merged into one algorithm to get much more pertinence and harmony in searching process, thus providing a viable way for designing high-performance optimization algorithm.

Key words: evolutionary algorithm; clustering evolution optimization model; clustering searching mechanism; numerical optimization

进化算法(evolutionary algorithm,简称 EA)具有许多突出的优点,如良好的并行性、自适应性、鲁棒性、对目标函数的类型和搜索空间的形状没有任何限制等等.这些特点使得它的应用领域几乎涉及到所有传统方法难以解决的优化问题.总的来看,进化搜索的动力源自对"优胜劣汰,适者生存"这一自然法则的模拟,这使得它的基本计算框架既有良好的普适性,又具有与生俱来的随机盲目性.在实际运算中,EA 需要有效协调模拟进化系统中不同角度的多种具有对立性的运算过程,如基因对编码模式的勘探和开采、个体对问题空间的全局和局部搜索、群体对多样性和逼近性的平衡等,否则就易出现过早收敛或收敛缓慢等现象,并严重影响系统的计算

* 基金项目: 国家自然科学基金(61203311, 61105064); 陕西省教育厅科研计划(2013JK1183)
收稿时间: 2013-11-09; 修改时间: 2014-01-10; 定稿时间: 2014-03-28

效果.

自适应机制是 EA 提高自身协调性的一种较为常用的方法[1].但制定合理、有效的自适应策略本身就是一个难题,而且受计算模型复杂性的限制,自适应机制对系统性能的改进效果也有限.近年来的一些研究尝试通过引入其他学科中的相关理论或方法,从不同角度来改进 EA 的计算过程,如仿效玉米基因传递过程中出现的基因跳跃现象而提出的基因重组机制[2]、引入优化实验设计方法而提出的搜索策略[3,4]和受热力学自由能极小过程启发设计的选择模型[5]等等.由于这些改进机制都可从不同层面有效提高 EA 的计算效率和可靠性,因此上述研究在特定领域都获得了良好的效果.近期的另一个研究动态是通过融合多种异构的搜索机制来混合建模[6,7],此类研究显示:搜索机制的多样化不但可有效改善系统的计算性能,还可进一步提高求解问题的稳定性.但如何平衡系统建模中多样性与复杂性以及不同搜索机制的特殊性及其相互间协调性的关系,就成为了一个新问题.纵观上述研究可发现:EA 对系统运算过程的调节和控制能力对它的计算效果有着重要的影响,且其重要性随着算法模型的愈发复杂而越来越突出.

从生物学的角度看,许多重要进化事件的发生基础是基因聚类[8];而从社会学的角度看,部落、种族等具有类结构特征的社会单元则是人类进化和文明发展的基本组织.可见,具有簇类形态的组织结构在复杂群体进化的发生和发展过程中都起到了至关重要的调节和导向作用.受此启发,我们认为可在个体间构建一定结构的簇类组织,并利用该组织对模拟进化系统的运算过程进行控制和协调,以获得更为高效的进化搜索性能.基于此思想,提出了一种新型的群体搜索优化方法——类搜索算法(clustering searching algorithm,简称 CSA).本文介绍了 CSA 的基本计算模型,并设计出融合进化算子和差分进化(differential evolution,简称 DE)计算方法的类搜索算法——CSA/DE.对 6 个典型高维测试函数和 6 个复杂混合函数的仿真实验及分析表明:CSA/DE 可通过控制类组织的结构有效调节系统的运算状态,并基于类搜索在系统不同的计算阶段形成更具针对性且协调性更佳的搜索机制组合,实现对高纬连续问题稳定、可靠的优化计算.

本文第 1 节首先对相关的研究工作进行分析并针对所存在的问题提出类搜索模型.第 2 节分析簇类结构的改变对群体搜索特性的影响规律,同时提出一种基于层次聚类[9]的类组织构造方法.第 3 节介绍 CSA/DE 的主要计算机制.第 4 节是仿真实验及分析.最后进行总结和展望.

1 类搜索模型

1.1 相关工作及问题

在群体的进化搜索中,个体间的编码差异及其在问题空间的分布,与它们所参与搜索运算的性能特点间有着一定的联系,而研究者一直在尝试利用这类关系来调节群体的搜索过程.如:文献[10]通过基于编码相似性匹配的约束机制(mating restriction)来控制交叉运算中个体对的形成过程,以提高算法对多目标优化问题的求解性能.此类研究中,系统对个体间亲疏关系的利用主要集中于特定的搜索运算内.而另一些算法则希望通过控制个体在问题空间的分布形态来构成结构化群体,并基于此调节系统的整体计算过程.如:小生境机制[11]通过排挤或适应度共性策略来抑制群体的趋同过程并形成个体间一定程度的聚集,这种群体结构改善了系统对多峰问题的计算效果.但小生境的形成是一个间接且被动的过程,而且个体间的结构关系是一种隐式的存在,这使得算法不便于利用其对系统的计算过程进行更为主动的控制.近年的一些研究开始尝试对个体间的彼此关系进行特定的区分,并利用某种显式的类别结构来直接控制群体的搜索过程.如:文献[12]从性别的角度将个体分为父和母两个组,其中,母个体承担对编码空间的全局采样,而父个体则用于确定局部搜索的区域范围.系统通过对两组个体选择过程的控制来实现对全局和局部搜索过程的调节.另外,文献[13]利用混沌参数来量化个体间的差异,并根据个体所具有特征值的分布区间对群体进行分类,而参与差分计算的个体则挑选于不同子类.这一方法提高了 DE 对作业调度问题的优化效果.

上述研究使研究者认识到:对个体间亲疏关系的利用有助于改进算法的计算性能,但其计算效果依赖于系统对群体在问题空间中分布状态的有效分析.为了改进这一点,部分研究中引入了聚类计算,其中,

- 一些研究关注于通过聚类分析获得群体在问题空间中的分布特征并用于优化系统的控制参数.如文

献[14]利用 k 均值聚类运算对群体分类,并以拥有当前最优个体和最差个体的子类中所包含个体的数量差异为依据,通过一个模糊系统对交叉和变异概率进行动态调节;

- 而另一些研究则主要是针对差分进化算法的改进,聚类机制在 DE 中被用来计算群体所达到空间中的一些典型坐标,其作用类似于一个局部搜索算子.如:文献[15]通过 k 均值聚类将群体中的个体分为 k 个子类,并通过计算簇类的重心坐标来产生 k 个新个体;文献[16]则通过 k 个簇类重心个体进一步计算群体的重心矢量,并围绕其随机产生 $p–k$ 个新个体再与 k 个簇类重心个体一起组成规模为 p 的下代群体;文献[17]增加了两种交叉算子,可分别围绕 k 个簇类重心个体以及群体的重心个体来进行搜索;文献[18]也采用类似的搜索机制.

上述研究显示:建立在个体间的簇类结构有助于提升模拟进化系统的自我调节能力,改进群体的搜索性能.但上述研究中簇类的规模 k 及其初始结构多是随机产生的,而事实上,类组织的规模和结构与群体的搜索特性间有着紧密的联系,因此要使系统在整个搜索过程中能够持续保持合理、有效的计算状态,就需要动态地对簇类的规模和结构进行优化调整,这是现有研究所欠缺的.此外,上述研究中聚类的主要功能多是为了获得群体在问题空间的分布状态或者得到局部区域的代表性矢量,这未能充分利用类结构所形成的对局部空间进行求精搜索的有利条件.同时,簇类间较大的差异性也有助于提高群体对问题空间的勘探效率,这一点在相关研究中同样未被有效重视.另外,簇类结构也为算法协调系统中各种全局性和局部性计算机制间的关系提供了新的途径,而这一研究思路在现有工作中则未被提及.

1.2 类搜索模型及相关定义

针对上述研究中的不足,本文提出了一种以簇类为计算的核心单元并基于类结构驱动的新型进化搜索模型——类搜索算法(CSA),其基本计算框架如图 1 所示.在该模型中,类组织是一种依据群体在问题编码空间中的分布结构,通过聚类计算形成的个体连接关系.在搜索运算过程中,系统将完全由类组织来控制和驱动.

- 首先,搜索运算被区分为类间搜索和类内搜索两个层面的计算过程.

类间搜索通过不同子类间的协作完成对问题空间的勘探,而类内搜索则通过同一子类内的信息交互实现多个局部区域内并行的求精.这种分工机制不仅有利于降低系统中全局性和局部性搜索运算间的耦合性、为更细致地协调二者间的关系提供基础,同时也为有效融合具有不同计算特性的搜索机制提供了环境.

- 其次,系统将通过类迭代完成群体更替.

我们注意到,真实的自然选择过程是局部且动态的,但在传统的选择模型中,个体基于全局、静态的适应度评价易使个别性能突出的个体产生“占群”现象,促使群体早熟.在类迭代中,系统将以子类为单位分别独立进行选择运算,这样既可全面地对群体进行采样,也可保证代表性个体的局部竞争优势.因而,类迭代机制更符合自然选择的特点,并为实现群体多样性与逼近性间的平衡提供了一种新的方法.

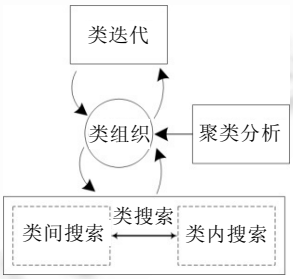


Fig.1 Model of CSA

图 1 类搜索模型

此外,相对于现有的相关研究,CSA 也具有一些独有的特点.

- 首先,簇类组织并不是多个物理上独立的子群体,而是建立在单一群体之上的一种虚拟的个体关系划

分,这是CSA与多种群协同进化算法[19]最本质的差别.CSA的这一结构特点,使得系统既可方便地实现信息共享,又可灵活地改变个体与子类的归属关系,避免了多种群结构中子群体间通信以及个体迁移所需的复杂计算;

- 其次,簇类组织的规模和结构不是随机的,而是通过聚类计算动态地进行有目的的调整和优化,聚类计算在 CSA 中的作用不是为了产生具有类内相似度最大或类间差异度最大的特定结构,而是通过对类组织的构造来调节系统的计算状态,故将聚类计算机制融入群体搜索系统的过程有其特殊性,并不是对某种聚类方法的简单移植;
- 再次,类组织改变了 EA 传统的系统搜索和控制模式.在 CSA 中,类组织是一种介于群体和个体之间且粒度可变的单位,这不同于传统的"个体-群体"静态的两级结构.类结构的这种可变性,使算法在自我调节能力上具有了一种类似调焦的特性,系统可通过放大或缩小子类的粒度以及类组织的规模来控制群体的搜索行为,并基于对类间和类内搜索过程间关系的协调来优化群体中全局性和局部性计算间的相互作用过程.

为了便于对算法进行描述,首先给出 CSA 在连续空间中的一些基本定义.

定义 1. 个体 $\boldsymbol{I}$ 是一个属于 D 维问题空间 S 的实数矢量,记为 $\boldsymbol{I}\in S$;群体 Pop 是规模为 p 的个体集合,即,$Pop=\{\boldsymbol{I}_1,\boldsymbol{I}_2,\ldots,\boldsymbol{I}_p\}$.其中,个体 $\boldsymbol{I}_i=x_1,x_2,\ldots,x_D$ 的适应度值为 $f_i>0$,且 $\boldsymbol{I}_i$ 中每一维变量 x_j 的搜索区间为

$$x_{low,j}\leqslant x_j\leqslant x_{up,j}, j=1,\ldots,D.$$

定义 2. 个体 $\boldsymbol{I}_i$ 与 $\boldsymbol{I}_j$ 间的相似性可通过欧氏距离比 Dis_{ij} 来度量:

$$Dis_{ij}=Dis(I_i,I_j)=\|I_i-I_j\|/\|x_{up}/x_{low}\| \tag{1}$$

其中,$\boldsymbol{x}_{up}$ 和 $\boldsymbol{x}_{low}$ 为 S 的上下界矢量,$\|\boldsymbol{x}-\boldsymbol{y}\|$表示 $\boldsymbol{x}$ 和 $\boldsymbol{y}$ 两个实数矢量的欧几里德范数.

显然,$Dis_{ij}\in[0,1]$,且 Dis_{ij} 越趋近于 0,说明 $\boldsymbol{I}_i$ 与 $\boldsymbol{I}_j$ 的相似性越大;反之,则正好相反.

定义 3. 群体中所有个体对$\{(\boldsymbol{I}_i,\boldsymbol{I}_j)|i\neq j,\boldsymbol{I}_i,\boldsymbol{I}_j\in Pop,i,j=1,2,\ldots,p\}$的平均欧氏距离比$\gamma$为

$$\gamma=\sum_{i=1}^{p-1}\sum_{j=i+1}^{p}Dis_{ij}\Big/C_p^2 \tag{2}$$

由上述定义可知:$\gamma\in[0,1]$,且γ越趋近于 0,说明群体的平均差异性越小;反之,则正好相反.故,可用其作为群体多样性和收敛性的观察指标.

定义 4. 类组织 C 是一种建立在个体间虚拟的动态连接关系,其中,子类 C_i 所包含的成员记为 A_i^j 并单映射于群体中的一个个体 $\boldsymbol{I}_k$,而中心成员 A_i^c 是 C_i 中具有最大适应度的成员.类组织结构可如下描述:

$$C=\{C_1,C_2,\ldots\},C_i=\{A_i^1,A_i^2,\ldots\},A_i^j\to I_k,I_k\in Pop,i,j=1,2,\ldots,k=1,2,\ldots,p \tag{3}$$

$$\varnothing=\{C_i\cap C_j\mid i\neq j\},\varnothing=\{A_i^r\cap A_j^s\mid i\neq j\},Pop=\bigcup C_i,i,j,r,s=1,2,\ldots \tag{4}$$

$$Dis(A_i^s,A_j^q)>Dis(A_i^s,A_i^r),i\neq j,i,j,r,s,q=1,2,\ldots \tag{5}$$

$$\forall A_i^r\in C_i,f_{A_i^r}\leqslant f_{A_i^c},i,r=1,2,\ldots \tag{6}$$

上述定义说明:子类中的任意成员都是群体中某个个体的唯一投影,而类组织则是依据成员所映射个体在问题空间中的分布特征形成的簇类化结构,故类组织的变化所改变的只是个体间虚拟的连接关系,个体在群体中的物理位置并未改变,这样有利于提高系统的聚类计算效率.另外,子类所属成员间无交集且类结构使得同一子类的成员间更具相似性,而不同子类的成员间更具差异性.同时,子类的中心成员成为了某个局部区域中最具竞争力和代表性的个体.

2 通过类结构调节群体的搜索状态

实现 CSA 需要解决的关键一点就是系统如何构造出能够改进当前群体搜索性能的类组织,首先,我们来讨论类组织结构的变化对系统计算性能的影响规律.

2.1 类结构与群体搜索特性间的联系

假设群体搜索仅使用基本的变异和交叉算子,由于变异运算的发生概率较小,故类组织主要是依靠交叉运算来对问题空间进行勘探,因此可主要针对该运算来进行讨论.设群体中的个体都不重复,那么由 $p(p>2)$个个体构成的群体可搜索空间 Φ_{Pop} 将是由 $C_p^2 = p(p-1)/2$ 个交叉搜索子域所组成.其中,由个体对$(\boldsymbol{I}_i,\boldsymbol{I}_j)$构成的交叉子域的大小可表示为 $\Phi(I_i,I_j)=\int_{I_j}^{I_i} \mathrm{d}\bar{X}$. 显然:$\boldsymbol{I}_i$ 和 $\boldsymbol{I}_j$ 间差异率越大,则所构成的搜索子域空间将越大,而交叉搜索所能覆盖的区域就越广,搜索运算就更倾向于全局搜索,但同时,搜索达到交叉子域中某个点的概率将越低;反之,交叉运算将具有较小的覆盖域和较高的域内到达概率,即,更倾向于局部搜索.由类组织的结构特点可知:类间搜索将更倾向于全局搜索,而类内搜索则属于局部搜索.

若设类组织 C 中所有子类内的个体对$(\{(\boldsymbol{I}_i,\boldsymbol{I}_j)|i\neq j,\boldsymbol{I}_i,\boldsymbol{I}_j\in C_k,k=1,2,\ldots,|C|\})$之间的平均欧氏距离比为 a,而所有子类间的个体对$(\{(\boldsymbol{I}_i,\boldsymbol{I}_j)|\boldsymbol{I}_i\in C_r,\boldsymbol{I}_j\in C_s,r\neq s,r,s=1,2,\ldots,|C|\})$的平均欧氏距离比为 b,且类内和类间个体对的数量分别为 x 和 y,则可得等式$\gamma=(x\cdot a+y\cdot b)/(x+y)$.由于在一代搜索运算中群体的平均欧氏距离比$\gamma$以及 $x+y=C_p^2$ 都是定值,故可设 $Y=C_p^2\gamma$,那么等式可变为

$$Y=x\cdot a+y\cdot b \tag{7}$$

显然:当每个个体构成一个子类,即$|C|=p$ 时,x 与 a 将不存在,而 $b=\gamma$且 $y=p$;若整个群体构成了一个子类,即$|C|=1$ 时,y 与 b 将不存在,而 $a=\gamma$且 $x=p$.这是类结构的两种极端情况,在这两种情况下,类结构实际等效于无结构的单一群体.而当 $1<|C|<p$ 时,由$|C|$的变化会引起 x 与 y 以及 a 与 b 的改变,从而引起系统运算状态的变化.故,通过调整类结构,就可控制交叉运算所倾向的全局或局部搜索的平均搜索子域规模及子域数量,从而调节系统对问题空间的全局覆盖率以及对局部可搜索域内各点的到达概率.

首先,可给出子类规模$|C|$影响类搜索特性的两个定理.

定理 1. 在满足定义(4)的类组织中,子类规模$|C|$的增大,将使得类内交叉子域数量 x 减少而类间交叉子域数量 y 上升;反之,则刚好相反.

证明:设$\Omega_1=\{\boldsymbol{\Phi}(\boldsymbol{I}_i,\boldsymbol{I}_j)|\boldsymbol{I}_i,\boldsymbol{I}_j\in C_i,i=1,2,\ldots,|C|\}$,$\Omega_2=\{\boldsymbol{\Phi}(\boldsymbol{I}_i,\boldsymbol{I}_j)|\boldsymbol{I}_i\in C_i,\boldsymbol{I}_j\in C_j$ and $i\neq j,i,j=1,2,\ldots,|C|\}$,且 $n_i=|C_i|$.显然:

$$p=n_1+n_2+\ldots+n_{|C|},C_p^2=|\Omega_1|+|\Omega_2|=C_{n_1}^2+C_{n_2}^2+\ldots+C_{n_{|C|}}^2+|\Omega_2|=\sum_{i=1}^{|C|}C_{n_i}^2+|\Omega_2|.$$

为便于分析,可令 $n_i=p/|C|$,那么,

$$|\Omega_1|=\sum_{i=1}^{|C|}C_{n_i}^2=|C|C_{p/|C|}^2=\frac{p^2}{2|C|}-\frac{p}{2},|\Omega_2|=\sum_{i=1}^{|C|-1}\sum_{j=2}^{|C|}(n_in_j)=\frac{|C|(|C|-1)p^2}{2|C|^2}=\frac{p^2}{2}\left(1-\frac{1}{|C|}\right).$$

可见:当$|C|$增大时,$x=|\Omega_1|$将减小而 $y=|\Omega_2|$将增大;反之,则正好相反.图 2(a)为随着$|C|$的变化所引起的类内交叉子域数量 x 以及类间交叉子域数量 y 变化的大体趋势.证毕. □

定理 2. 在满足定义(4)的类组织中,类内个体的平均欧氏距离比 a 和类间个体的平均欧氏距离比 b 以及群体平均欧氏距离比γ总满足 $0\leqslant a\leqslant\gamma\leqslant b\leqslant 1$,且子类规模$|C|$的降低将使得 a 和 b 都呈现上升趋势,反之则相反.

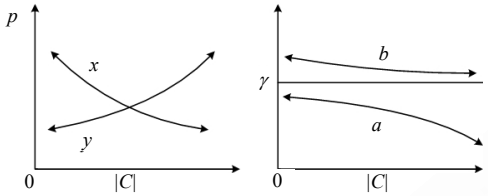
证明:由类组织的结构特点可知 $0\leqslant a\leqslant\gamma$,而用 b 除以γ可得 $\frac{b}{\gamma}=\frac{x}{y}\left(\frac{\gamma-a}{\gamma}\right)+1\geqslant 1$,因此,$a$ 与 b 总满足 $0\leqslant a\leqslant\gamma\leqslant b\leqslant 1$.另外,由聚类计算的特点可知:在$|C|$从 p 趋于 1 的过程中,a 呈现的是从 0 趋于γ的变化过程.下面将主要讨论 b 的变化规律.若一次子类的合并使得类结构从 C^1 变为了 C^2,则显然$|C^1|>|C^2|$.假设在这个过程中类间个体对减少了 t 个,其欧氏距离比分别为 $Dis_1,Dis_2,\ldots,Dis_t$,且 C^1 和 C^2 中的类内和类间的个体对数量以及平均欧氏距离比分别为 x_1 与 y_1、a_1 与 b_1 以及 x_2 和 y_2 与 a_2 和 b_2.那么由公式(7)可得:

$$Y=x_1\cdot a_1+y_1\cdot b_1=x_2\cdot a_2+y_2\cdot b_2=(x_1\cdot a_1+Dis_1+Dis_2+\ldots+Dis_t)+(y_1-t)\cdot b_2.$$

假设 t 个欧氏距离比的均值为 h,对上式化简可得 $y_1\cdot b_1-t\cdot h=(y_1-t)b_2$.由于类合并是发生在距离最小的子类间,故该子类的所属个体与其余个体都具有相对较小的间距,即,满足 $h<b_1$.于是可得到以下关系:

$$(y_1-t)\cdot b_1<y_1\cdot b_1-t\cdot h=(y_1-t)b_2.$$

所以有 $b_1<b_2$.可见,随着$|C|$的降低,b将逐步增大.图2(b)所示为随着类组织收敛促使类内和类间平均欧氏距离比都升高这一过程的大体趋势.证毕. □



(a) (b)

Fig.2 Variation trend of a, b and x, y with $|C|$

图2 x与y以及a与b随$|C|$的变化趋势

综合定理1和定理2,可总结出类结构的变化影响群体搜索状态的一般规律:子类规模$|C|$的增大会增加类搜索中类间交叉子域的数量,使群体倾向全局搜索,但类间交叉子域平均规模的降低则会影响群体对问题空间勘探覆盖的效果,所以,$|C|$过大反而会影响系统的全局搜索效率;而子类规模$|C|$的减小会增加类搜索中类内交叉子域的数量,使群体倾向局部搜索,但类内交叉子域平均规模的升高也会影响群体局部到达的速度,所以,$|C|$过快地收敛同样会影响系统局部求精的效率.因此,在迭代搜索中,CSA 为提高类组织搜索效率,需要动态地寻找类结构的平衡点,不断地优化类间和类内搜索运算间的关系.

2.2 基于层次聚类构造类组织

为了便于对类组织的结构进行调节,需要一种可在无监督状态下完成群体聚类的方法.层次聚类可由数据底部开始向上透过一种层次架构反复将数据进行聚合,最终形成一个层次序列的聚类问题解[9].由于这个层次序列的形成过程有利于对类的规模和结构进行调节,故我们在层次聚类基本思想的基础上发展出了类组织的构造方法,其基本步骤为:以单个个体为子类的最小初始单位,并在一个迭代过程中不断合并间距最小的两个子类来改变类组织的结构,直到满足某个终止条件为止.显然,这一过程中影响类结构的关键因素是子类聚合的终止条件.我们注意到:当最小子类间距超过γ时,若继续进行类合并会使类间交叉子域的数量出现过大幅度的降低,同时使类内交叉子域的平均规模过大.这样既会影响类间搜索对问题空间的全局覆盖率,也会影响类内求精的效率.因此,类结构中的最小子类间距不宜大于γ.另外,群体的进化搜索过程使得γ随着群体收敛产生了一个逐渐递减的趋势,因此,若依据γ的变化来调节类组织的结构可以自然地产生一个类组织规模逐渐收敛的过程,并使类组织形成由面向全局搜索到逐渐过渡到局部搜索的类结构动态演变.故,子类聚合的终止条件可设定为

$$\min[D(C_i,C_j)]>\gamma,C_i,C_j\in C,C_i\neq C_j \tag{8}$$

此外,当群体收缩到一个较小区域时,较高的个体相似性使群体已局限于局部搜索,此时可直接将群体置为一个子类,并通过单纯的类内搜索来加速群体收敛.在实际运算过程中,个体间欧氏距离比的标准差δ被用作判断类组织是否需要完全收敛的依据,具体公式如下:

$$\delta=\sqrt{\sum_{i=2}^{2p}\sum_{j=1}^{j<i}(idm_{ij}-\gamma)\Big/C_{2p}^2}<\delta_{\min} \tag{9}$$

其中,$\delta_{\min}$是一个设定的较小值.

为了提高计算效率,设置了矩阵IDM与CDM来分别存储个体间距idm_{ij}和类间距cdm_{ij}.

设$idm_{ij}\in IDM,cdm_{ij}\in CDM$:

$$idm_{ij}=\begin{cases}0, & \text{if } i\leqslant j\\ Dis_{ij}, & \text{else}\end{cases},i,j=1,2,...,2p \tag{10}$$

$$cdm_{ij}=\begin{cases}0, & \text{if } i\leqslant j\\ Dis(C_i,C_j)=\sum_{I_r\in C_i}\sum_{I_s\in C_j}idm_{rs}\Big/|C_i||C_j|, & \text{else}\end{cases}\quad (i=1,2,...,j=1,2,...) \tag{11}$$

显然:矩阵 *IDM* 与 *CDM* 都是下三角矩阵,且其对角线上的元素都为 0.设类搜索将产生 p 个新个体,利用 *IDM* 与 *CDM* 对父代及其子代全部 $2p$ 个个体进行层次聚类分析并构造类组织的过程如下所示:

算法 1. 类组织的层次聚类构造方法.

步骤 1. 通过公式(10)初始化 *IDM*,并计算当前群体的平均欧氏距离比γ以及标准差δ,判断公式(9)是否成立:若成立,则将整个群体置为一个子类并结束聚类计算;否则,执行步骤 2.

步骤 2. 将每个个体 $\boldsymbol{I}_i(i=1,2,\ldots,2p)$都置为一个子类 C_i,并由 *IDM* 初始化 *CDM*.

步骤 3. 从 *CDM* 中找到间距最小的两个子类 C_r 和 C_k,即 $cdm_{rk}=\min(CDM)$,若满足 $cdm_{rk}<\gamma$,则执行步骤 4;否则,结束聚类计算.

步骤 4. 合并 C_r 与 C_k 组成为新子类 C'_{ik},接着,通过公式(11)更新 *CDM* 中 C'_{ik} 与其余子类的间距值,并返回步骤 3.

3 融合差分计算机制的类搜索进化算法(CSA/DE)

类搜索模型中,类间和类内搜索相互独立的结构使我们可以方便地将具有不同计算结构和运算特性的搜索方法有效地结合在一起.但本文主要讨论的是类搜索机制的作用,故在 CSA/DE 中,类间搜索依然沿用了传统的“交叉+变异”的算子组合,而类内搜索则引入了差分搜索计算中典型的“DE/best/1/bin”策略.图 3 为 CSA/DE 的实现流程,其中,群体聚类基于算法 1 实现,而初始化、类搜索和类迭代的计算机制介绍如下.

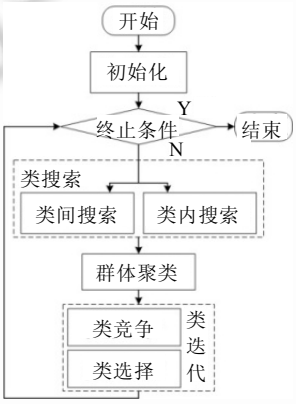


Fig.3 Flowchart of CSA/DE

图 3 CSA/DE 的计算流程

3.1 初始化

CSA 的初始化包括群体初始化和类组织初始化两部分.在群体初始化中,系统将在问题空间 S 中对群体进行随机分布,个体 $\boldsymbol{I}_i=x_{i,1},x_{i,2},\ldots,x_{i,D}$ 的初始化公式为

$$x_{i,j}=x_{low,j}+UR(0,1)\cdot|x_{up,j}-x_{low,j}|,j=1,2,\ldots,D,i=1,2,\ldots,p \tag{12}$$

其中,$UR(0,1)$是一个在(0,1)区间均匀分布的随机数.

在类组织初始化中,初始群体中的每个个体将被置为一个初始子类以及该子类的中心成员,并形成规模为 p 的初始类组织,即:

$$C_i=\{A_i^1\},A_i^c=A_i^1,A_i^1\to I_i,i=1,2,...,p \tag{13}$$

子类 C_i 的控制序号 i 取自个体 $\boldsymbol{I}_i$ 在 *Pop* 中的序号,此外,控制子类 C_i 搜索规模的参数 τ_i 被统一置为 2.

3.2 类搜索

类搜索的主要计算过程如图4中的算法2所示,其中,参数 ch 为本代类搜索产生的新个体数量,参数 $\eta\in(0,1)$ 被用来调节类内搜索的发生概率,而系统将利用子类序号 i 依次控制不同子类进行搜索.参数 τ_i 量化了子类 C_i 的竞争力,它在类搜索中的具体作用是限定 C_i 允许产生的后代个体数量,它使得竞争力较强的子类可以更高的频率参与搜索运算.该值在初代中由初始化产生,之后将在前代的类选择过程中计算获得.首先进行的类间搜索将以子类为单位完成广域勘探,所产生的新个体将加入当前子类 C_i.接着,竞争力相对较高的子类(即,中心个体适应度值大于群体平均适应度值的子类)有机会通过类内搜索提高其所属成员的竞争力.

算法 2. 类搜索.
```
i=1;
ch=0;
while (ch<p)
{
  t=0;
  while (t<τi)        //Ci的搜索过程
  {Ci=SearchingAmongCluster(Ci,C);
    if (f_Aic > f̄ && UR(0,1) > η)
      SearchingInsideCluster(Ci)
    t=t+2;
  }
  ch=ch+t;
  i=i+1;
}
```

Fig.4 Core steps of clustering searching

图 4 类搜索的主要步骤

3.2.1 类间搜索(SearchingAmongCluster)

以交叉运算为核心的类间搜索需要有两个个体参与.

- 当子类规模满足|C|>1 时,在子类 C_i 的类间搜索操作过程中,将首先从类组织 C 中随机选择一个子类 $C_j(i\neq j)$,接着从 C_i 和 C_j 中各随机挑选出一个成员 $A_i^r=x_1,x_2,\ldots,x_D$ 与 $A_j^s=y_1,y_2,\ldots,y_D$;
- 若|C|=1,即,当类组织中仅存在一个子类时,则仅从 $C_i(i=1)$中随机挑选出成员 $A_i^s=x_1,x_2,\ldots,x_D(A_i^s\neq A_i^c)$,而另一个所需成员 $A_i^r=y_1,y_2,\ldots,y_D$ 将选择为群体当前最优个体,即 $A_i^r=A_i^c$.

完成个体挑选后,将通过交叉和变异操作产生两个新个体 $\boldsymbol{I}'=x_1',x_2',\ldots,x_D'$ 和 $\boldsymbol{I}''=x_1'',x_2'',\ldots,x_D''$,计算公式如下:

$$I',I''=\begin{cases}x_k'=NM(x_k),x_k''=NM(y_k), & \text{if } UR(0,1)<\beta_m\\ x_k'=u\cdot x_k+(1-u)\cdot y_k,x_k''=u\cdot y_k+(1-u)\cdot x_k, & \text{if } UR(0,1)<\beta_c\ (k=1,2,\ldots,D)\\ x_k'=x_k,x_k''=y_k, & \text{otherwise}\end{cases} \tag{14}$$

其中,β_m 为变异概率,交叉概率 $\beta_c\in(\beta_m,1)$,$u=UR(0,1)$,NM 为非均匀变异操作.

类间搜索中的交叉运算在类规模|C|的影响下实际具有两种模式:当|C|>1 时是面向广域勘探的“类间交叉”,而当|C|=1 时则是具有邻域搜索特征的“类内交叉”.另外,非均匀变异的具体公式为

$$NM(x_k)=\begin{cases}x_k+\Delta(g,x_{up,k}-x_k), & \text{if } U(0,1)\geqslant 0.5\\ x_k-\Delta(g,x_k-x_{low,k}), & \text{else}\end{cases} \tag{15}$$

其中,g 为当前的进化代数,而函数 $\Delta(g,y)$ 可返回一个 $(0,y)$ 区间的数值.我们希望随着 g 趋于最大迭代次数 G 而逐渐增大 $\Delta(g,y)$ 趋于 0 的概率,故采用下式来计算 $\Delta(g,y)$ 的值:

$$\Delta(g,y)=y\cdot(1-u^{(1-g/G)}) \tag{16}$$

3.2.2 类内搜索(SearchingInsideCluster)

子类 C_i 的类内搜索是只发生在其所属成员内的局部搜索运算.基于“DE/best/1/bin”策略,系统首先需要从 C_i 中随机挑选出成员 $A_i^j=x_1,x_2,\ldots,x_D$ 以及 A_i^r 和 A_i^s $(r\neq s)$,接着由子类中心成员 A_i^c 以及 A_i^r 和 A_i^s 通过差分计算产

生一个随机矢量 $\boldsymbol{v}=v_1,v_2,\ldots,v_D$.

$$v = A_i^c + F\cdot(A_i^r - A_i^s) \tag{17}$$

其中,$F\in(0.3,0.9)$.之后,由 A_i^j 与 $\boldsymbol{v}$ 交叉产生一个零时个体 $\boldsymbol{I}'=x_1',x_2',\ldots,x_D'$.

$$I' = \begin{cases} x_k' = v_k, & \text{if } UR(0,1) < CR \\ x_k' = x_k, & \text{else} \end{cases} (k=1,2,\ldots,D) \tag{18}$$

其中,$CR\in(0.3,0.9)$.最后,判断个体 $\boldsymbol{I}'$的适应度值:若其大于成员 A_i^j 的适应度值,则用 $\boldsymbol{I}'$替换成员 A_i^j 在群体中所映射的个体;否则,将放弃 $\boldsymbol{I}'$.

3.2.3 类搜索的调节机制

算法 2 中,类内搜索的概率为 $1-\eta$,故η的取值将直接影响系统在每代进行的搜索频度.在 CSA/DE 中,参数η将利用一个模拟退火过程来进行调节,以下是计算公式:

$$\eta=(\exp(-\gamma_g/T_g)-1)\cdot\xi+1 \tag{19}$$

其中,γ_g 为第 g 代群体的平均欧氏距离比,而第 g 代群体的退火温度 $T_g=T_0\cdot\rho^{\,g}$,T_0 为总迭代次数 G;ρ是一个略小于 1 的数,$\xi\in(0,1)$.

另外,类间搜索中变异运算的发生概率β_m将通过一个线性函数来调节.第 g 代群体的变异概率为

$$\beta_m = \beta_m^u\cdot(1-g/G)+\beta_m^l \tag{20}$$

其中,G 为总迭代数,$\beta_m^u,\beta_m^l\in(0,0.1)$.

图 5 为 $T_0=G=200$,$\rho=0.95$,$\xi=0.95$ 且γ分别为 0.5,0.2 和 0.05 时,参数η所呈现出的变化曲线.由公式(19)可知,CSA/DE 的类内搜索发生概率为 $1-\eta=\exp[(-\gamma_g/T_g)-1]\xi$.而由图 5 可见:参数$\eta$的变化,使得类内搜索的发生概率 $\exp[(-\gamma_g/T_g)-1]\xi$保持了逐渐增大的趋势,并可随着群体多样性的降低适当放缓趋于最大值ξ的速度.另外,公式(20)则使得β_m随着迭代数 g 的增大逐渐从 $\beta_m^u+\beta_m^l$ 递减到 β_m^l.上述调节机制使得类搜索在系统计算前期形成了由“类间交叉+变异”主导的以子类为单位的全局性搜索;而在计算中期则经过了一个由“类间交叉+变异+DE/best/ 1/bin”混合而成的从全局性搜索逐渐向局部性搜索转换的过渡阶段;随着类组织逐渐收敛为 1,在系统计算后期最终形成了由“类内交叉+DE/best/1/bin”为主体的邻域搜索.

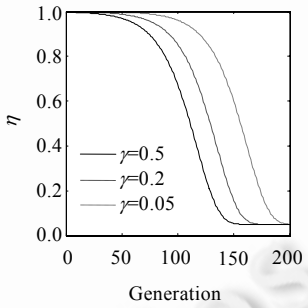


Fig.5 Parametric curve of η

图 5 参数η的变化曲线

3.3 类迭代

类搜索过程结束后,系统将基于算法 1 完成对当前群体及其后代个体的空间分布分析,并产生新的类结构.在类迭代运算过程中,系统将通过类竞争和类选择来构造下代类组织及群体.这一过程中,子类 C_i 的竞争力可通过其中心个体的适应度值来衡量,并具体表现为参数τ_i,该值将确定子类 C_i 是否能出现在下代类组织中,以及可存活个体的数量.同时,τ_i 也将作为 C_i 在下代类搜索中产生后代个体的数量限制值.

3.3.1 类竞争

算法 3. 类竞争.

步骤 1. 对类组织中的所有子类按适应度值对其所属成员进行排序,得到序列 $C_i=\{A_i^1,A_i^2,...\}$,再将 A_i^1 置为该子类的中心成员 A_i^c.

步骤 2. 若|C|=1,则置 $\tau_1=p$,并结束类竞争计算;若|C|>1,则执行步骤 3.

步骤 3. 首先,依据子类中心成员的适应度值对类组织 C 进行排序,并以子类在队列中的位置序号 i(i=1,2,…,|C|)作为子类 C_i 的控制序号;接着,计算所有子类中心成员的平均适应度 f' 以及满足 $f_{A_i^c}>f'$ 的子类数量 s.此外,使用参数 lp 记录可分配给其余子类的剩余个体数量,并置参数初始值 lp=p,i=1.

步骤 4. 对子类 C_i,计算其权重值 h_i.

$$h_i=\begin{cases}2(s-i+1)/(s^2+s), & \text{if } i\leqslant s\\ 0, & \text{else}\end{cases} \tag{21}$$

计算子类 C_i 的参数 τ_i.

$$\tau_i=K(h_i\cdot p,lp) \tag{22}$$

其中,函数 K 先将表达式 $h_i\cdot p$ 的计算结果向上取整为一个正偶数 n,即,n=0,2,4,6,…

- 当 n=0 时,$\tau_i=0$;
- 当 n>0 时,
  - ✓ 若 lp>0,当 $lp\geqslant n$ 时,$\tau_i=n$;而当 lp<n 时,$\tau_i=lp$;
  - ✓ 若 $lp\leqslant 0$,则 $\tau_i=0$.

步骤 5. $lp=lp-\tau_i$,$i=i+1$,若 $i\leqslant|C|$,则返回步骤 4;否则,将退出.

3.3.2 类选择

类选择通过对子类以及子类内所属个体的两个层面的选择过程完成群体更替.首先进行子类选择,只有满足参数 $\tau_i>0$ 的子类将可生存到下代类组织中;接着,对子类内的成员进行个体选择.

算法 4. 类选择.

步骤 1. 对子类 C_i,i=1,2,…,|C|,若 $\tau_i=0$,则子类 C_i 将被删除,其所属个体都将被淘汰,$i=i+1$;若 $\tau_i>0$,则执行步骤 2.

步骤 2. 若 $|C_i|>\tau_i$,则截取序列中前 τ_i 个成员组成下代子类 C_i',并使其成员所映射个体进入下代群体;若 $|C_i|<\tau_i$,则先将序列中的成员加入 C_i',并使所映射个体进入下代群体,其余 $\tau_i-|C_i|$ 个成员所映射的下代群体中的个体将通过对中心成员 A_i^c 的变异获得,即:

$$C_i'=\{A_i^1,A_i^2,...,A_i^{|C_i|+1}=NM(A_i^c),...,A_i^{\tau_i}=NM(A_i^c)\} \tag{23}$$

最后,将具有最优适应度的成员置为 C_i' 的中心成员 A_i^c.

步骤 3. 将子类 C_i' 加入下代类组织 C',$i=i+1$,返回步骤 1.

4 仿真实验

使用 VC++6.0 实现了 CSA/DE,并在 PC(2.4GHz/2GRAM)上对 6 个典型的多维测试函数[7]和 6 个复杂的混合测试函数[20]进行了仿真实验.

4.1 对典型测试函数的优化实验

多维测试函数 f_1~f_4 和 f_5,f_6 分别选自文献[7]中的多模态函数 f_1~f_4 和单模态函数 f_{11} 与 f_{13}.在对 30 维函数 f_1~f_6 的优化实验中,CSA/DE 将与 RCCRO[21],GAAPI[7]以及 OEA[22]这 3 种近年来提出的具有代表性的算法进行比较.RCCRO 对 f_2 的评估次数限定为 250 000,对其余函数的评估次数都限定为 150 000[21].GAAPI 的终止条件是蚁巢对同一区域的重复搜索次数不超过 30[7].OEA 的终止条件是每个函数的最大评估次数为 300 000[22].CSA/DE 采用的终止条件是搜索 2 000 代,群体规模 p=60,类间搜索中变异概率调节参数 $\beta_m^u=1/D$,$\beta_m^l=1/2D$.根据公式(20),变异概率将从 3/2D 向 1/2D 逐步递减.另外,交叉概率为 0.9,类内搜索中 F=0.4,CR=0.8.在构造 η 的

相关参数中,T_0=2000,ξ=0.99,而ρ对 6 个函数的取值分别为 0.97,0.96,0.97,0.96,0.95 和 0.95.在聚类分析过程中,$\delta_{\min}$ 对 6 个函数的取值依次为 0.02,0.02,0.01,0,0.01 和 0.01.

表 1 为 CSA/DE 对 f_1~f_6 独立进行 50 次运算的实验统计结果与上述 3 种算法实验数据的对比.就计算结果的平均质量(MFV)来说:CSA/DE 对函数 f_1 的计算精度与 OEA 相当,对函数 f_4 和 f_6 则获得了最佳质量的解;而 OEA 对其余 3 个函数的计算结果则最优,CSA/DE 次之.从总的个体搜索数量,即适应度函数的平均评估次数(MNFE)来看,OEA 的运算量最大,基本在 300 000 次,GAAPI 的运算量最小,大体为 OEA 的 1/3 左右,而 CSA/DE 与 RCCRO 的运算量适中,基本为 OEA 的一半.该实验结果说明,基于类搜索模型的模拟进化系统在对高维连续问题的优化计算中具有较高的竞争力.该结果表明了 CSA 的可行性.

Table 1 Comparison of experimental results between CSA/DE and 3 algorithms for f_1~f_6 with 30 dimensions

表 1 CSA/DE 与 3 种算法对 30 维函数 f_1~f_6 优化结果对比

Function		f_1	f_2	f_3	f_4	f_5	f_6
CSA/DE	MNFE	152 916	160 200	154 810	152 863	162 856	163 911
	MFV (Std)	**−12 569.486 6 (5.97e−12)**	3.46e−10 (1.96e−9)	1.25e−8 (2.85e−8)	**1.54e−16 (2.04e−16)**	4.92e−10 (2.73e−10)	**3.02e−16 (2.06)e−16**
RCCRO	MNFE	150 000	250 000	150 000	150 000	150 000	150 000
	MFV (Std)	−1.257e+4 (2.317 e−2)	9.077e−4 (2.876e−4)	1.944e−3 (4.190e−4)	1.117e−2 (1.622e−2)	6.427e−7 (2.099e−7)	2.196e−3 (4.341e−4)
GAAPI	MNFE	26 510	24 714	19 592	29 647	29 199	30 714
	MFV (Std)	−12569.4 (2.68e−1)	5.046e−3 (7.4e−3)	3.89e−2 (5.45e−2)	7.78e−2 (2.25e−1)	1.07e−2 (1.76e−2)	5.58e−2 (5.46e−2)
OEA	MNFE	300 019	300 019	300 018	300 020	300 017	300 014
	MFV (Std)	**−12 569.486 6 (5.555e−12)**	**5.430e−17 (1.683e−16)**	**5.336e−14 (2.954e−13)**	1.317e−2 (1.561e−2)	**2.481e−30 (1.128e−29)**	2.068e−13 (1.440e−12)

MNFE: Mean number of function evaluation; MFV: Mean function value; Std: Standard deviations

4.2 对混合测试函数的优化实验

为了更真实地模拟现实问题所具有的复杂结构,文献[20]提出通过对函数进行坐标位移、正交旋转以及空间叠加等一系列措施来生成更具挑战性的混合测试函数 Cf_1~Cf_6.同时,文献[20]中还记录了 PSO,CPSO,CLPSO,CMA-ES,G3-PCX 和 DE 这 6 种算法对混合函数的优化结果.这 6 种算法对各函数都进行了 20 次运算,且对每个函数进行优化时,最大适应度函数评估次数被限制为 50 000 次.此外,HEM[23]对 Cf_1~Cf_6 分别进行 20 次优化,其中,对 Cf_1 和 Cf_3 的最大迭代次数是 300 代,且个体数量的波动范围在 600~100 之间;对 Cf_2 以及 Cf_4~Cf_6 的最大迭代次数是 500 代,且个体数量的波动范围在 1 000~150 之间.文献[24]对 Cf_1~Cf_6 分别进行 30 次计算,并使最大函数评估次数等于 50 000 次.

为了使最大适应度函数评估次数保持在 50 000 次左右,CSA/DE 的群体规模 p 被设置为 40,而最大迭代数为 500.同时,对 6 个混合函数也分别进行了 20 次计算,其中,交叉概率、变异概率调节参数以及差分计算参数都与第 4.1 节的实验相同.在参数η的构造函数中,T_0=500,ξ=0.99,ρ的取值都为 0.95.由于混合函数结构复杂,问题空间中存在大量的局部最优点,故群体极易出现早熟.为了提高 CSA/DE 的抗早熟能力,对聚类分析参数$\delta_{\min}$ 都置为 0.表 2 为 CSA/DE 对 6 个混合函数的实验统计结果.在 20 次计算中,CSA/DE 对 Cf_1 和 Cf_3 所获得的最小值都接近 0,对 Cf_2 和 Cf_5 所获得的最小值在个位数,对 Cf_4 和 Cf_6 所获得的最优值则较大.此外,除了对 Cf_3 计算的方差较大外,对其余函数优化结果的波动幅度都较小.

Table 2 Experimental statistic results of CSA/DE for Cf_1~Cf_6

表 2 CSA/DE 对混合函数 Cf_1~Cf_6 的优化统计结果

Function	Function value				MNFE	Function	Function value				MNFE
	Max	Min	Mean	Std			Max	Min	Mean	Std	
Cf_1	1.70e−7	6.58e−9	5.8e−8	6.1e−7	55 218	Cf_4	110.0	94.7	103.8	4.8	53 789
Cf_2	16.0	3.86	7.58	8.0	53 830	Cf_5	5.97	1.85	3.5	1.2	54 681
Cf_3	168.1	6.1e−8	45.1	63.6	54 297	Cf_6	303.9	301.3	302.2	0.93	54 152

表 3 为 CSA/DE 与文献[20,24]中运算结果相对突出的 CLPSO,DE 和 CSO+CSO-OED(I)以及文献[23]中的 HEM 优化统计均值数据的比较.对 Cf_3,Cf_4 和 Cf_6,CSA/DE 的计算结果质量最高;对 Cf_1,CSO+CSO-OED(I)的计算精度最佳,CSA/DE,CLPSO 以及 HEM 的计算精度接近并都好于 DE;而对 Cf_2 和 Cf_5,HEM 的计算结果最优,CSA/DE 的计算结果则要略好于其他 3 种算法.整体上看:在个体搜索运算量基本相当的情况下,CSA/DE 相对上述算法对 6 个混合函数都体现出了较好的计算效果.该实验进一步说明:类搜索模型不但是可行的,而且是有效的,通过对类结构和类搜索机制的动态调节,CSA/DE 可对具有不同复杂结构的问题空间获得高效、稳定的搜索性能.

Table 3 Comparison of experimental results between CSA/DE and 4 algorithms for Cf_1~Cf_6

表 3 CSA/DE 与 4 种算法对混合函数优化结果的对比

Function		Cf_1	Cf_2	Cf_3	Cf_4	Cf_5	Cf_6
CSA/DE	MFV	5.8e−8	7.58	**45.1**	**103.8**	3.5	**302.2**
	Std	6.1e−7	8.0	**63.6**	**4.8**	1.2	**0.93**
CLPSO	MFV	5.74e−8	19.16	132.81	322.32	5.371	501.16
	Std	1.04e−7	14.75	20.03	27.46	2.61	0.78
DE	MFV	0.07	28.76	144.41	324.86	10.79	490.94
	Std	0.11	8.63	19.40	14.75	2.60	39.64
HEM	MFV	4.39e−8	**0.12**	77.63	240.83	**1.82**	495.08
	Std	6.46e−9	**0.53**	69.78	24.44	**1.35**	22.26
CSO+CSO-OED(I)	MFV	**2.80e−13**	1.47e+1	1.52e+2	3.26e+2	8.52e+0	5.05e+2
	Std	**1.52e−13**	2.97e+0	2.18e+1	1.99e+1	1.69e+0	2.99e+1

4.3 CSA/DE的计算机制分析

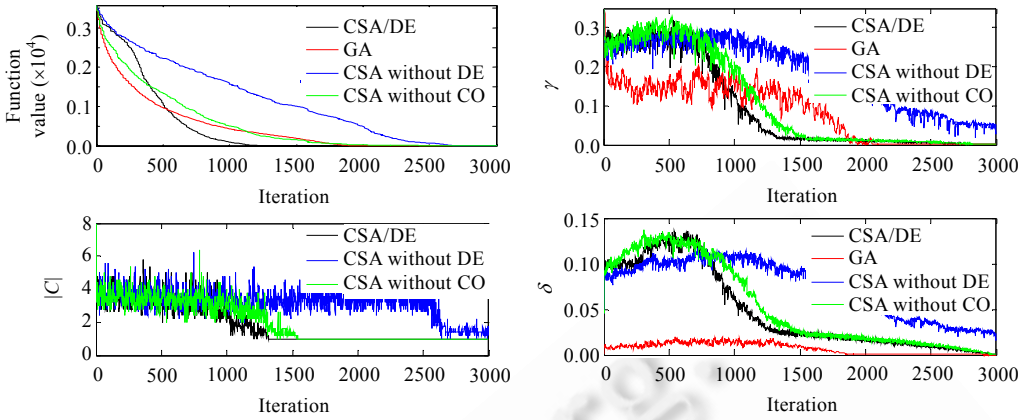
在本节中,我们将通过实验数据的对比来讨论类结构以及类内搜索和类搜索调节机制对 CSA/DE 计算性能的影响.首先,使用 CSA/DE、遗传算法(GA)和去掉类内搜索的 CSA/DE——CSA without DE(将 CSA/DE 中的参数 η 置为 1,使类内搜索的发生概率为 0)以及去掉类搜索调节机制的 CSA/DE——CSA without CO(将 CSA/DE 中的参数 η 置为定值 0.5)对 100 维函数 f_1~f_6 分别进行 20 次独立运算(对 f_1 进行了调整,以使其最优函数值为 0,$f_1(x)=\sum_{i=1}^{D}(-x_i\sin\sqrt{|x_i|})-D\times 418.98288727243369, S=[-500,500]^D$).实验中,CSA/DE 的群体规模 p=60,总迭代数为 3 000 代,交叉概率、变异概率调节参数、差分计算参数以及聚类分析参数 $\delta_{\min}$ 都与第 4.1 节的实验相同.参数 η 的构造函数中,T_0=3000,ξ=0.99,ρ 的取值都为 0.98.CSA without DE 的相关参数都与 CSA/DE 相同.CSA without CO 中,变异概率 β_m=0.03,其他运算和控制参数都与 CSA/DE 相同.GA 中,群体规模 p=100,迭代数量也为 3 000 代,这样可使其最大函数评估次数大于其他算法.同时,GA 使用算术交叉和非均匀变异进行搜索,并基于线性排序选择完成群体迭代,其中,交叉和变异公式以及控制参数都与 CSA without CO 相同.表 4 为上述算法的计算统计结果.

Table 4 Experimental statistic results of 4 algorithms for f_1~f_6 with 100 dimensions

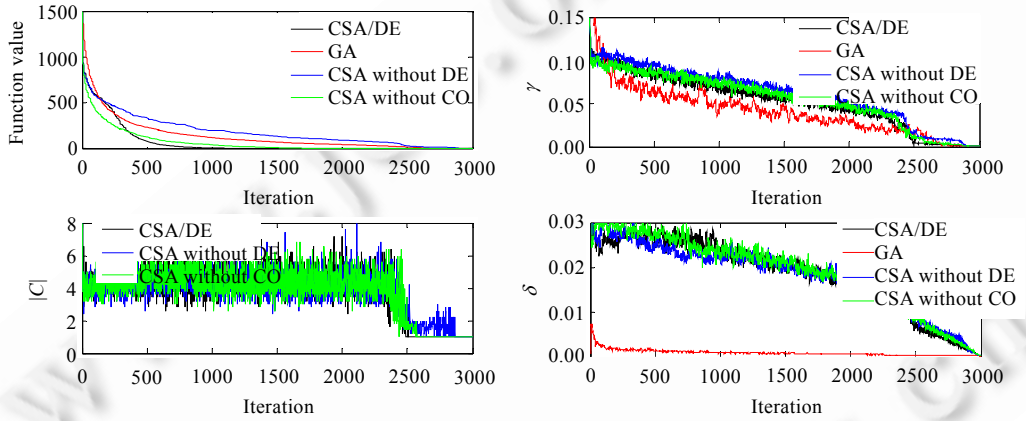
表 4 4 种算法对 100 维函数 f_1~f_6 优化统计结果

Function	Algorithm	MFV	Std	MNFE	Function	Algorithm	MFV	Std	MNFE
f_1	CSA/DE	8.74e−10	3.00e−10	258 962	f_2	CSA/DE	1.17e−7	1.23e−7	256 656
	GA	4.88e−2	1.91e−2	300 000		GA	3.81e−2	4.22e−2	300 000
	CSA without DE	1.26e−1	2.52e−1	187 967		CSA without DE	6.97e−3	3.20e−2	192 538
	CSA without CO	1.84e−6	2.72e−7	224 378		CSA without CO	7.36e−7	1.44e−7	229 511
f_3	CSA/DE	4.76e−6	9.40e−5	244 341	f_4	CSA/DE	1.42e−8	1.90e−8	275 198
	GA	5.20e−2	3.11e−2	300 000		GA	6.41e−1	4.57e−1	300 000
	CSA without DE	1.38e−2	8.09e−3	210 241		CSA without DE	2.43e−2	1.57e−2	220 749
	CSA without CO	3.43e−4	1.16e−4	230 628		CSA without CO	3.30e−5	3.71e−5	249 102
f_5	CSA/DE	5.56e−6	5.27e−6	260 346	f_6	CSA/DE	1.46e−14	1.32e−14	270 318
	GA	8.37e−1	2.46e−1	300 000		GA	1.36e−1	3.51e−1	300 000
	CSA without DE	1.96e−2	1.26e−2	207 573		CSA without DE	8.48e−2	3.73e−2	208 686
	CSA without CO	1.50e−4	3.71e−5	241 085		CSA without CO	6.62e−10	7.18e−10	242 769

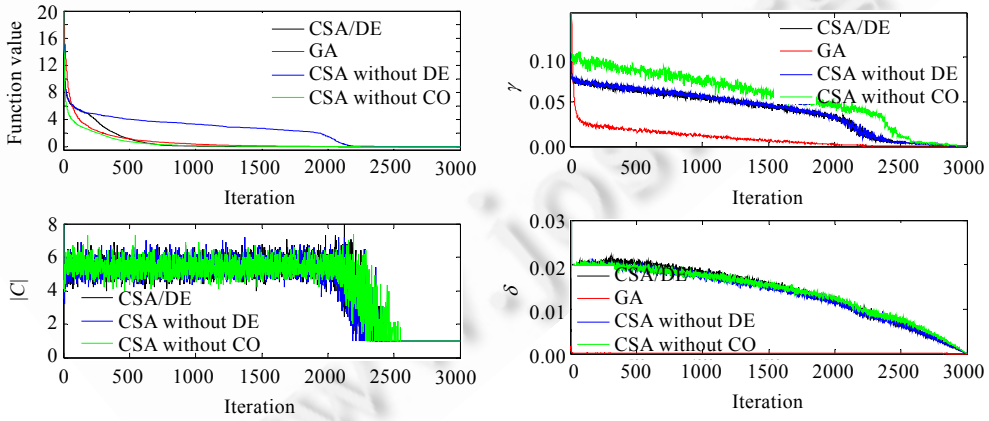
图 6 为 4 种算法在迭代搜索过程中优化函数值(function value)、类规模(|C|)、群体平均欧氏距离比(γ)及标准差(δ)这 4 个参数的均值变化曲线.



(a) f_1



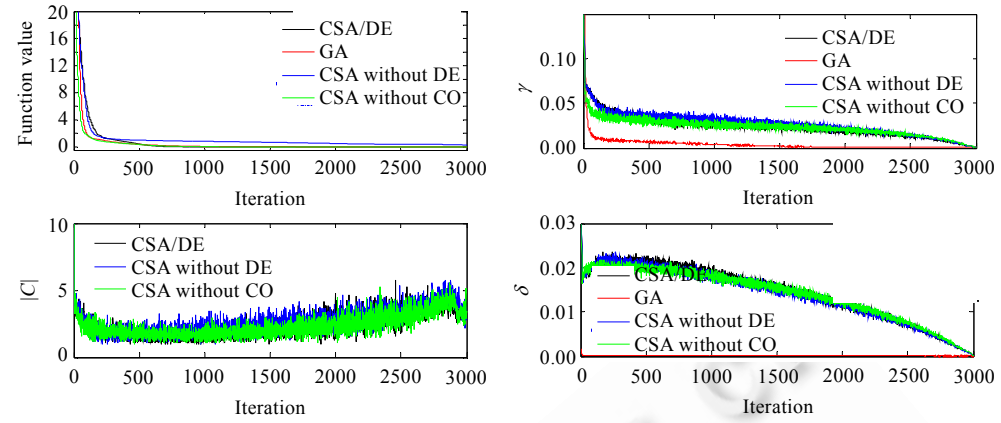
(b) f_2



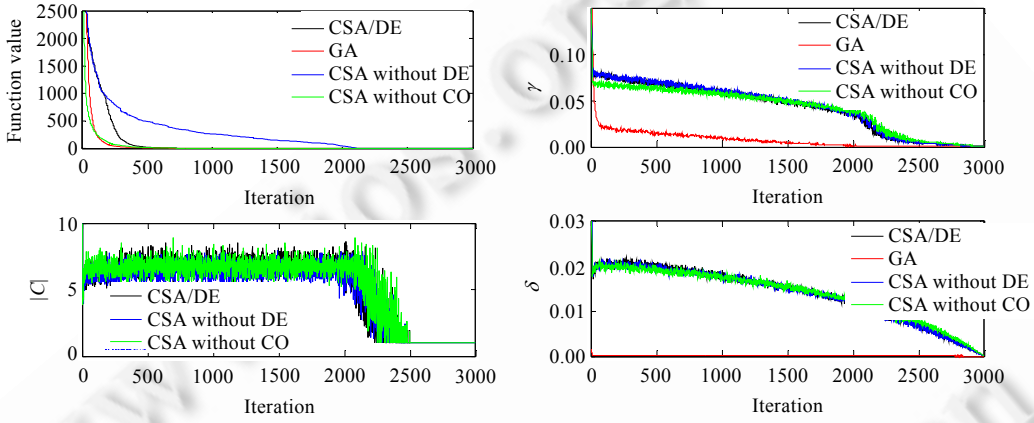
(c) f_3

Fig.6 Parameters variation curve of 4 algorithms for f_1~f_6 with 100 dimensions

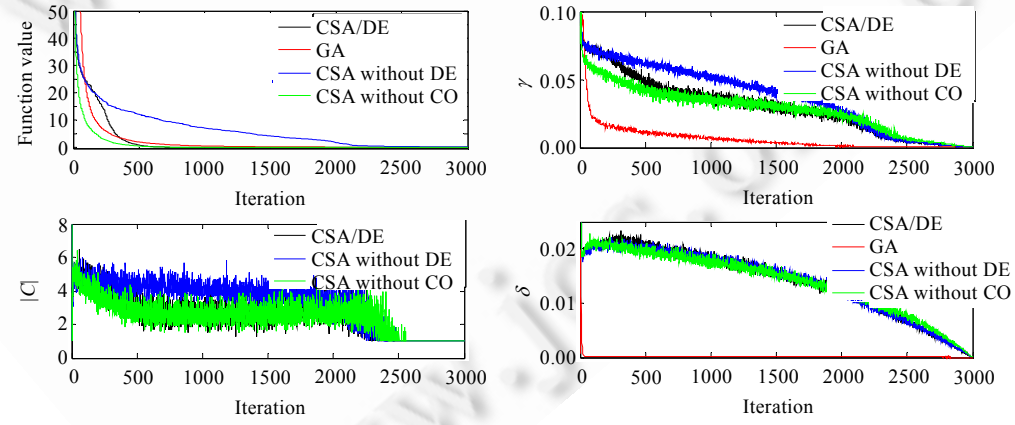
图 6 4 种算法对 100 维函数 f_1~f_6 的参数变化曲线



(d) f_4



(e) f_5



(f) f_6

Fig.6 Parameters variation curve of 4 algorithms for f_1~f_6 with 100 dimensions (contiuned)

图 6 4 种算法对 100 维函数 f_1~f_6 的参数变化曲线(续)

CSA without DE 与 GA 都仅使用传统的交叉和变异算子搜索新个体,但对比表 4 中 CSA without DE 与 GA 的优化统计结果可以发现:除 f_1 外,CSA without DE 对其余函数的优化质量都要稍好于 GA,且前者的个体搜索

运算量只为 GA 的 2/3 左右.而对比图 6 中的函数值变化曲线可以发现二者间搜索过程的差异:计算前期,GA 的函数值曲线的下降速度明显都要快于 CSA without DE,但在靠近全局最优函数值时开始明显放缓;而 CSA without DE 函数值曲线的前期下降速度虽然较慢,但相对而言,却能更为匀速、稳定地逼近最优值.此外,表 4 中 GA 的计算精度也要明显弱于 CSA/DE 和 CSA without CO(统称 CSA).可见:类结构有利于抑制群体过快收敛产生的进化停滞,保证群体更稳定地向全局最优点逼近.另外,CSA without DE 的统计数据和函数值曲线显示:单纯的类间搜索不能有效提高系统的收敛速度,但利用“DE/best/1/bin”策略所具有的保优和局部信息交换的特点,则可实现类内搜索和类间搜索有益的互补.

图 6 中,群体的平均欧氏距离比γ反映了群体交叉运算对问题空间的平均覆盖率,而标准差δ则反映了群体交叉中子域规模与平均值的差异率.也就是说:δ越大,则群体交叉运算中子域规模的变化就越丰富;反之,则更一致.这两个参数的变化规律可从一个侧面反映出类结构对群体搜索性能的影响.从图 6 中类规模($|C|$)的变化可以发现:除 f_4 中由于$\delta_{\min}$=0 抑制了类组织收敛外,CSA 对其余函数的计算中,$|C|$都基本保持了一个稳定递减并收敛为 1 的趋势.当$|C|$>1 时,CSA 中群体的平均欧氏距离比γ的下降速度要明显慢于 GA;而当类组织收敛为 1 时,群体则开始加速收敛,其收敛速度甚至要快于 GA.这一现象在函数 f_1 和 f_2 的曲线图中表现得最为清晰.CSA 与 GA 中γ的变化差异说明:一定的类结构使群体维持了更为稳定的个体差异,这使系统保持了对问题空间更有效的全局勘探性能.而类组织的收敛则可有效加速群体收敛,使群体更高效地集中于某个局部区域的求精.此外,图 6 中,GA 的标准差δ往往会迅速接近 0;而 CSA 中,δ随$|C|$保持稳定递减的趋势则非常明显.GA 中,δ接近 0 说明群体迭代搜索过程中交叉运算的性能差异区分不明显,倾向全局或局部的搜索运算间具有较高的耦合性.而 CSA 中,δ稳定的递减过程则使得系统更便于区分全局性和局部性的搜索运算,也就有利于系统在整个搜索过程中协调二者间的分工.

综上可见,类结构有利于群体抑制过早收敛,但系统的整体运算性能取决于类间搜索和类内搜索的有效协调.而 CSA/DE 能够在计算的不同阶段形成互补的搜索机制组合,构成更具针对性的运算模式,从而也就实现了计算量的合理分配,并获得理想的计算效果.

5 结 论

本文介绍了类搜索模型(CSA)的基本概念和定义,并设计出融合传统进化搜索算子和差分计算机制的函数优化算法 CSA/DE.大量的仿真实验结果说明,CSA/DE 是一种对多维连续问题高效、稳定的搜索优化方法.上述工作一方面验证了 CSA 的可行性和有效性,另一方面则显示:利用类搜索模型可有效融合具有不同计算特性的搜索机制,并动态协调彼此间的作用过程,形成有效的互补.因此,基于 CSA,可根据不同问题空间的结构特点设计更具针对性且协调性更佳的搜索优化系统.在下一步工作中,我们将针对 CSA 的这一特性尝试将其应用于对具体优化问题的指导.

References:

[1] Wang RF, Jiao LC, Liu F, Yang SY. Nature computation with self-adaptive dynamic control strategy of population size. Ruan Jian Xue Bao/Journal of Software, 2012,23(7):1760−1772 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/4151.htm [doi: 10.3724/SP.J.1001.2012.04151]

[2] Chan M, Man KF, Kwong S, Tang KS. A jumping gene paradigm for evolutionary multiobjective optimization. IEEE Trans. on Evolutionary Computation, 2008,12(2):143−159. [doi: 10.1109/TEVC.2007.895269]

[3] Cai ZX, Jiang ZY, Wang Y, Luo YD. A novel constrained optimization evolutionary algorithm based on orthogonal experimental design. Chinese Journal of Computers, 2010,33(5):855−864 (in Chinese with English abstract). [doi: 10.3724/SP.J.1016.2010.00855]

[4] Subbaraj P, Rengaraj R, Salivahanan S. Enhancement of self-adaptive real-coded genetic algorithm using taguchi method for economic dispatch problem. Applied Soft Computing, 2011,11(1):83−92. [doi: 10.1016/j.asoc.2009.10.019]

[5] Ying WQ, Li YX, Sheu PCY, Wu Y, Yu FH. Geometric thermo dynamical selection for evolutionary multi-objective optimization. Chinese Journal of Computers, 2010,33(4):755−767 (in Chinese with English abstract). [doi: 10.3724/SP.J.1016.2010.00755]

[6] Wang L, Li LP. A coevolutionary differential evolution with harmony search for reliability–redundancy optimization. Expert Systems with Applications, 2012,39(5):5271−5278. [doi: 10.1016/j.eswa.2011.11.012]

[7] Ciornei I, Kyriakides E. Hybrid ant colony-genetic algorithm (GAAPI) for global continuous optimization. IEEE Trans. on Systems, Man, and Cybernetics—Part B, 2012,42(1):234−245. [doi: 10.1109/TSMCB.2011.2164245]

[8] Song G, Zhang LX, Vinar T, Miller W. CAGE: Combinatorial analysis of gene-cluster evolution. Journal of Computational Biology, 2010,17(9):1227−1242. [doi: 10.1089/cmb.2010.0094]

[9] Tang XQ, Zhu P. Hierarchical clustering problems and analysis of fuzzy proximity relation on granular space. IEEE Trans. on Fuzzy Systems, 2013,21(5):814−824. [doi: 10.1109/TFUZZ.2012.2230176]

[10] Ishibuchi H, Narukawa K, Tsukamoto N, Nojima Y. An empirical study on similarity-based mating for evolutionary multiobjective combinatorial optimization. European Journal of Operational Research, 2008,188(1):57−75. [doi: 10.1016/j.ejor.2007.04.007]

[11] Ling Q, Wu G, Yang Z. Crowding clustering genetic algorithm for multimodal function optimization. Applied Soft Computing, 2008,8(1):88−95. [doi: 10.1016/j.asoc.2006.10.014]

[12] García-Martíneza C, Lozanob M, Herrerab F, Molinac D, Sánchezd AM. Global and local real-coded genetic algorithms based on parent-centric crossover operators. European Journal of Operational Research, 2008,185(3):1088−1113. [doi: 10.1016/j.ejor.2006.06.043]

[13] Davendra D, Zelinka I, Bialic-Davendra M, Senkerik R, Jasek R. Clustered enhanced differential evolution for the blocking flow shop scheduling problem. Central European Journal of Operations Research, 2012,20(4):679−717. [doi: 10.1007/s10100-011-0198-3]

[14] Zhang J, Chung H, Lo W. Clustering-Based adaptive crossover and mutation probabilities for genetic algorithms. IEEE Trans. on Evolutionary Computation, 2007,11(3):326−335. [doi: 10.1109/TEVC.2006.880727]

[15] Cai ZH, Gong WY, Ling CX, Zhang H. A clustering-based differential evolution for global optimization. Applied Soft Computing, 2011,11(1):1363−1379. [doi: 10.1016/j.asoc.2010.04.008]

[16] Wang YJ, Zhang JS, Zhang GY. A dynamic clustering-based differential evolution algorithm for global optimization. European Journal of Operational Research, 2007,183(1):56−73. [doi: 10.1016/j.ejor.2006.10.053]

[17] Liu G, Li YX, Niea X, Zheng H. A novel clustering-based differential evolution with 2 multi-parent crossovers for global optimization. Applied Soft Computing, 2012,12(2):663−681. [doi: 10.1016/j.asoc.2011.09.020]

[18] Cai YQ, Wang JH, Yin J. Learning-Enhanced differential evolution for numerical optimization. Soft Computing, 2012,16(2):303−330. [doi: 10.1007/s00500-011-0744-x]

[19] Potter MA, De Jong KA. Cooperative coevolution: An architecture for evolving coadapted subcomponents. Evolutionary Computation, 2000,8(1):1−29. [doi: 10.1162/106365600568086]

[20] Liang JJ, Suganthan PN, Deb K. Novel composition test functions for numerical global optimization. In: Proc. of the 2005 IEEE Int'l Swarm Intelligence Symp. Pasadena: IEEE Computer Society Press, 2005. 68−75. [doi: 10.1109/SIS.2005.1501604]

[21] Lam AYS, Li VOK, Yu JJQ. Real-Coded chemical reaction optimization. IEEE Trans. on Evolutionary Computation, 2012,16(3):339−353. [doi: 10.1109/TEVC.2011.2161091]

[22] Liu J, Zhong WC, Jiao LC. An organizational evolutionary algorithm for numerical optimization. IEEE Trans. on Systems, Man, and Cybernetics—Part B: Cybernetics, 2007,37(4):1052−1064. [doi: 10.1109/TSMCB.2007.891543]

[23] Montiel O, Castillo O, Melin P, Díaz AR, Sepúlveda R. Human evolutionary model: A new approach to optimization. Information Sciences, 2007,177:2075−2098. [doi: 10.1016/j.ins.2006.09.012]

[24] Yu H, Jiao LC, Gong MG, Yang DD. Clonal selection function optimization based on orthogonal experiment design. Ruan Jian Xue Bao/Journal of Software, 2010,21(5):950−967 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/3472.htm [doi: 10.3724/SP.J.1001.2010.03472]

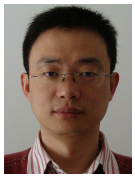
附中文参考文献:

[1] 王蓉芳,焦李成,刘芳,杨淑媛.自适应动态控制种群规模的自然计算方法.软件学报,2012,23(7):1760−1772. http://www.jos.org.cn/1000-9825/4151.htm [doi: 10.3724/SP.J.1001.2012.04151]

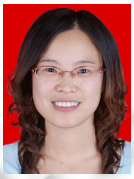
[3] 蔡自兴,江中央,王勇,罗一丹.一种新的基于正交实验设计的约束优化进化算法.计算机学报,2010,33(5):855−864. [doi: 10.3724/SP.J.1016.2010.00855]

[5] 应伟勤,李元香,Sheu PCY,吴昱,余法红.演化多目标优化中的几何热力学选择.计算机学报,2010,33(4):755−767. [doi: 10.3724/SP.J.1016.2010.00755]

[24] 余航,焦李成,公茂果,杨咚咚.基于正交实验设计的克隆选择函数优化.软件学报,2010,21(5):950−967. http://www.jos.org.cn/1000-9825/3472.htm [doi:10.3724/SP.J.1001.2010.03472]



陈皓(1978－),男,河北安新人,博士,副教授,CCF 会员,主要研究领域为进化计算,工程优化.



潘晓英(1981－),女,博士,副教授,CCF 会员,主要研究领域为进化计算,多智能体系统.