文章编号: 1001-0920(2019)08-1663-09 DOI: 10.13195/j.kzyjc.2018.0007

求解约束优化问题的新型帝国竞争算法

雷德明, 操三强†, 李 明

(武汉理工大学 自动化学院, 武汉 430070)

摘 要: 针对约束优化问题,提出一种约束处理的新策略,运用字典序方法同时优化问题的目标函数和约束违背程度,设计一种新型帝国竞争算法. 该算法给出成本和归一化成本的新定义,以避免殖民国家势力为零,并应用嵌入殖民地间全局搜索的同化、基于优秀殖民地的革命、殖民国家的差分进化和新型帝国竞争等策略提高求解质量. 基于两组约束优化标准测试函数的实验结果和算法对比表明,结合字典序方法的新型帝国竞争算法在约束优化问题的求解方面具有较强的优势.

关键词: 约束优化问题; 字典序方法; 帝国竞争算法; 同化; 革命

中图分类号: TP18 **文献标志码:** A

An imperialist competitive algorithm for solving constrained optimization problem

LEI De-ming, CAO San-qiang†, LI Ming

(School of Automation, Wuhan University of Technology, Wuhan 430070, China)

Abstract: To solve a constrained optimization problem, a new strategy is proposed, in which the lexicographical method is used to simultaneously optimize the objective function and the degree of constraint violation. A novel imperialist competitive algorithm (ICA) is presented, in which, cost and normalized cost are redefined to guarantee that the power of all imperialists exceeds zero, and some strategies such as the global search of colonies in assimilation, excellent colonies based revolution, differential evolution of imperialists and a new approach of imperialist competition are applied to improve solution quality. Many experiments are conducted based on two groups of test functions, and the ICA is compared with some algorithms from literature. The computational results show that the ICA with the lexicographical method has promising advantages for solving constrained optimization problems.

Keywords: constrained optimization problem; lexicographical method; imperialist competitive algorithm; assimilation; revolution

0 引 言

约束优化问题(constrained optimization problem, COP)是优化领域中的一个重要问题[1],广泛存在于工程实际且求解困难,其研究具有重要的实际和理论意义. COP一般有决策变量上下界约束条件以及等式/不等式约束条件. 按照约束条件的数学特性,还可以分为线性/非线性约束条件. 由于约束条件的存在,导致在决策变量的搜索空间中产生了不可行域. 从无约束到约束优化,优化任务从单纯的目标优化,变成同时考虑目标函数的优化和约束条件的处理两个方面,导致有效的无约束优化方法难以直接用来处理COP[2]. 约束处理策略对求解COP的算法具有重要的影响,近年来,研究者提出了大量将约束条件结合到智能算法的约束处理策略[3]. 目前,这些策略可以划分为6类: 罚函数法[4]、ε可行性法则[5]、随机排序法[6]、约束处理法[7]、多目标优化法[8-9]和混合法[10-13]. 其中,Cai等[8]提出了多目标优化方法,将COP转化为两目标优化问题,一个目标为问题的目标函数,另一个目标为约束违背程度. 甘敏等[11]提出了一种多目标优化与自适应惩罚的混合约束优化进化算法. Gao等[14]将COP转化为两目标问题并提出一种双种群协调进化差分算法. He等[15]采用一种可行性法则提出了一种混合粒子群算法. Wang等[16]利用共同进化的概念自适应调节罚因子的罚函数法提出了共同进化的粒子群算法. Fang等[17]和He等[18]针对资源约束工程调度问题提出了两种分布估计算

收稿日期: 2018-01-02; **修回日期:** 2018-03-13.
基金项目: 国家自然科学基金项目(61573264, 71471151).
责任编委: 王凌.
†通讯作者. E-mail: caosanqiang93@163.com.

法.

帝国竞争算法(imperialist competitive algorithm, ICA)是一种模拟社会政治行为的智能算法,已成功应用于函数优化[19-20]、 设备布局[21-22]、 生产调度[23-28]、装配线平衡[29-30]和旅行商问题[31]等领域,但较少应用于COP的求解. 根据文献[32],ICA既具有较强的邻域搜索能力,又是有效的全局优化方法,结构灵活,且不必像遗传算法(GA) 那样要加强局部搜索才能实现全局搜索与局部搜索的协调,因此,探讨ICA在COP求解方面的能力和优势具有现实意义.

多目标优化处理COP时,将COP转化为两目标问题后,COP的最优解只是两目标问题Pareto前沿上的一点,在应用约束智能算法逼近Pareto前沿(front)的过程中,有很多解对解决COP意义不大,甚至无意义,这样必然带来计算资源的浪费,降低算法的搜索效率. 为此,提出一种字典序方法,直接逼近Pareto前沿上COP的最优点,在此基础上,设计一种新型ICA,给出成本和归一化成本的新定义以避免殖民国家势力为零. 在同化过程中嵌入殖民地之间的全局搜索以增强多样性,同时采用基于优秀殖民地的革命、殖民国家的差分进化和新型帝国竞争策略以获得高质量的解. 最后,使用一组约束优化标准测试函数,并与两种算法进行对比,计算结果表明,结合字典序方法的新型ICA在COP求解方面具有较强的优势.

1 COP描述及约束处理新策略

一般情况下,COP可描述如下:

$$\begin{aligned}&\min\ y=F(x),\ x=(x_1,x_2,\cdots,x_n)\in R^n.\\&\text{s.t. } g_i(x)\leqslant 0,\ i=1,2,\cdots,l;\\&\quad h_j(x)=0,\ j=l+1,\cdots,p.\end{aligned}\tag{1}$$

其中:$x=(x_1,x_2,\cdots,x_n)\in R^n$为$n$维决策变量,$F(x)$为目标函数,$g_i(x)$为第$i$个不等式约束条件,$l$为不等式约束条件个数,$h_j(x)$为第$j$个等式约束条件,$p$为等式约束条件个数. 通常,等式约束$h_j(x)=0$转化为不等式约束条件[33-34]:$|h_j(x)|-\delta\leqslant 0$,其中$\delta$为等式约束的容忍参数,可定义为特定值,本文定义为0.000 05.

解x违反第j个约束条件的违反程度表示为

$$G_j(x)=\begin{cases}\max\{0,g_j(x)\},\ 1\leqslant j\leqslant l;\\ \max\{0,|h(x)|-\delta\},\ l+1\leqslant j\leqslant p.\end{cases}\tag{2}$$

解x总的约束违反程度为

$$G(x)=\sum_{j=1}^{p}G_j(x).\tag{3}$$

多目标优化法[8]将COP转化为目标是$F(x)$和$G(x)$的双目标优化问题,在搜索过程中,需要对整个搜索空间逐层搜索,找到Pareto前沿;然后在可行域内,COP退化成一个单目标优化问题,最优的Pareto前沿与纵轴的交点即为全局最优解,如图1所示.

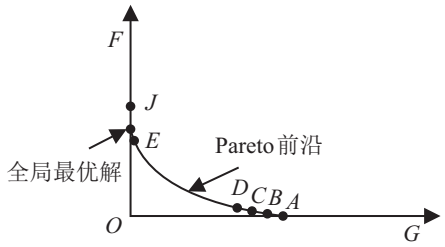

图1 多目标优化求解COP的最优解和Pareto前沿

显然,算法在逼近Pareto前沿过程中所产生的非劣解(如图1中的A,B,C,D)距离虚轴较远,违背约束的程度较严重,但它们在多目标优化过程中,往往比距虚轴较近的点E,甚至在虚轴上的解J对逼近Pareto前沿更有利;另外,由于最优解只是Pareto前沿的一个点,为了得到一个点而逼近整个前沿,必然带来计算资源的浪费,导致计算效率低下.

为此,引入字典序方法,具体描述如下:对于双目标问题$(F(x),G(x))$的两个解x,y,如果$G(x)<G(y)$或者$G(x)=G(y)$且$F(x)<F(y)$,则x优于y. 算法在搜索过程中,所产生的解先不断向虚轴靠近,直到到达虚轴,而一旦解x位于虚轴上即可行,则任何与x对比的解只有可行才可能优于x,这样算法可以沿虚轴不断逼近COP的全局最优解,同时显著地减少计算资源的浪费. 另外,字典序方法非常适合ICA这样的算法,ICA中没有选择操作,新旧种群的更替主要通过个体x和利用个体x产生的新解之间的比较来进行.

2 ICA原理描述

ICA是一种模拟人类殖民竞争过程的智能算法,由Atashpaz-Gargari等[35]于2007年提出,其主要过程描述如下.

Step 1: 初始化帝国. 随机生成一定数量的初始国家,将成本最小的几个国家定义为殖民国家,然后根据其自身势力大小为每个殖民国家分配相应数量的殖民地,构建初始帝国.

Step 2: 同化和革命. 在每个帝国内,对殖民地执行同化和革命.

Step 3: 殖民国家更新. 比较帝国内的殖民地和殖民国家,成本最小的国家成为新的殖民国家.

Step 4: 帝国竞争. 根据帝国的总成本,将最差帝国的最弱殖民地重新分配到新的帝国.

Step 5: 帝国消亡. 如果一个帝国内无任何成员, 则直接剔除这个帝国.

Step 6: 如果终止条件成立, 则搜索结束, 否则转至Step 2.

ICA中, 种群为所有国家的集合, 每个国家对应问题的一个解. 解$x=(x_1,x_2,\cdots,x_n)\in R^n$中分量$x_i$表示一个国家的文化、语言、经济或宗教等因素, 解的优劣根据其成本(cost)决定, 通常要求成本越小, 解越优, 对应的国家势力越大.

初始帝国构建过程中, 殖民国家的归一化成本定义为$C_k=\max\limits_l\{c_l\}-c_k$, 其中$C_k$表示殖民国家$k$的成本, 然后计算殖民国家的势力$p_k=\left|C_k\Big/\sum\limits_{l=1}^{N_{im}}C_l\right|$以及每个殖民国家所能分配的殖民地数量. 同化过程可以描述为殖民地沿其自身和殖民国家的连线方向向殖民国家随机移动一段距离$y\sim U(0,\beta\times d)$. 其中: d为殖民地和殖民国家之间的距离; $1<\beta\leqslant 2$, 设置$\beta>1$是为了使殖民地向殖民国家方向移动; $U(a,b)$表示区间(a,b)上的均匀分布. 然而, 同化后殖民地的新位置并不总在殖民地与殖民国家之间的连线上, 而是在与其连线大小为θ角度的射线上, 其中$\theta\sim U(-\varphi,\varphi)$.

革命主要针对少数殖民地进行某种改变, 其作用与GA的变异类似, 可以增强探索能力和避免早熟. 帝国竞争是ICA的重要步骤, 通过帝国竞争, 势力弱的帝国逐渐失去殖民地, 而势力强的帝国所拥有的殖民地越来越多.

GA和ICA的对比如表1所示.

表 1　GA与ICA对比

算法	操作	相似	不同
GA	交叉		种群中两个个体随机交换某些基因
	变异		对种群个体某些基因值做变动
	选择	交叉与同化: 产生新个体主要方式	优胜劣汰
ICA	同化	变异与革命: 对少数个体进行某种改变	殖民地向殖民国家靠拢
	革命		殖民地模拟某种非预期改变
	帝国竞争		弱国殖民地向强帝国聚集

3　求解COP的新型ICA

如上所述, 初始帝国构建时必然存在至少一个殖民国家, 其归一化成本和势力均为0, 从而无法为其分配任何殖民地; 同化主要发生在殖民地与殖民国家之间, 很少考虑殖民地向帝国内最好的殖民地学习, 革命时往往随机选择参加革命的殖民地; 另外, 殖民国家是种群内的优秀个体, 它们对算法的搜索具有引导作用, 如果殖民国家更新不及时, 则导致帝国内国家相似度增加, 降低种群多样性, 而过于频繁的帝国竞争会导致所有国家很快都聚集在一个帝国内, 从而可能引起算法早熟. 根据ICA中可能存在的上述问题和现象, 提出一种新型ICA, 并应用于COP的求解. 下面详细描述新型ICA的主要步骤.

3.1　初始帝国构建

初始帝国构建过程中, 由于约束项$G(x)$的加入, ICA中适合于单目标优化问题的成本以及归一化成本的计算公式不再适用, 为此, 采用如下方法计算成本: 确定初始种群内可行解的个数θ, 如果$\theta=0$, 即种群内所有解都不可行, 则$c_i=G(x_i)$; 否则, 根据下式计算c_i:

$$c_i=\begin{cases}F(x_i),\ G(x_i)=0;\\ 1+G(x_i)+F_{\max},\ \text{Otherwise.}\end{cases}\tag{4}$$

其中: $F_{\max}$为种群内可行解的最大目标函数值, c_i为解x_i的成本.

通常, 殖民国家的归一化成本$C_k=\max\limits_l\{c_l\}-c_k$, 这样至少会有一个殖民国家的归一化成本为0, 势力也为0, 导致这些殖民国家不能分配到任何殖民地, 相应的初始帝国由于没有殖民地而难以进行同化和革命, 从而影响ICA的搜索效率. 为此, 给出归一化成本的新定义, 有

$$C_k=2\times\max_l\{c_l\}-c_k.\tag{5}$$

这样可有效地避免归一化成本和势力为0, 所有的殖民国家都能分到一定数量的殖民地. 初始帝国构建过程如下:

Step 1: 计算所有国家的成本值, 选择成本最小的N_{im}个国家作为殖民国家.

Step 2: 利用式(5)计算殖民国家的归一化成本, 并计算前$N_{im}-1$个国家的势力p_k.

Step 3: 针对前$N_{im}-1$个殖民国家, 计算殖民地数量$NC_k=\text{round}(N_{\text{col}}\times p_k)$, 并为每个殖民国家随

机分配殖民地,剩下的殖民地直接分配给最后一个殖民国家.

其中round(·)给出最接近实数的整数,$N_{\rm col} = N - N_{im}$为殖民地总数.

3.2 同 化

通常,ICA同化过程没有殖民地之间的全局搜索,同化只是帝国内的所有殖民地与帝国内的殖民国家之间的全局搜索. 殖民地通过与殖民国家间的全局搜索,继承了殖民国家的优良信息,如果殖民国家较长时间内保持不变,则导致帝国内的国家相似度不断提高. 如果殖民地不仅可以向殖民国家学习,而且还能向帝国内优秀的殖民地学习,则既能避免帝国内解的多样性的降低,又能增大产生更优秀殖民地,甚至殖民国家的可能性. 为此,在同化过程中,加入了殖民地之间的全局搜索.

由于模拟二进制交叉(SBX[36])在函数优化中应用广泛并且效果良好,通过实验对比也发现,基于SBX的同化比常规的同化对提高ICA的性能更有利,利用SBX实现同化过程. SBX详细描述如下:对于解x_i, x_j,有

$$y_{1,k} = [(1+\beta)x_{i,k} + (1-\beta)x_{j,k}]/2, \tag{6}$$

$$y_{2,k} = [(1-\beta)x_{i,k} + (1+\beta)x_{j,k}]/2. \tag{7}$$

其中:$x_{i,k}, x_{j,k}$为x_i, x_j的第k个分量;u为[0, 1]区间的随机数;η为参数,实验中取$\eta = 1$;β为

$$\beta = \begin{cases} (2u)^{1/(\eta+1)}, & u \leqslant 0.5; \\ (1/2(1-u))^{1/(\eta+1)}, & u > 0.5. \end{cases} \tag{8}$$

同化只是产生新的殖民地,参与同化的殖民国家和优秀殖民地保持不变,这样每次只需产生$y_{1,k}$或$y_{2,k}$中的一个,为此,采用如下不同于GA的方式执行SBX:随机产生随机数u,如果$u < \alpha$,则根据式(6)计算$y_{1,k}$,否则,计算$y_{2,k}$,其中$\alpha = 0.5$. 同化过程的详细步骤如下.

Step 1: 运用字典序方法确定帝国中最优秀的殖民地η,针对每个殖民地λ执行Step 2和Step 3.

Step 2: 产生[0, 1]区间的随机数u,如果$u < P_a$,则对殖民国家与殖民地λ执行SBX,否则,若殖民地λ不是最优秀殖民地η,则对殖民地η和λ执行SBX,若是,则对殖民国家与殖民地λ执行SBX,产生新解z.

Step 3: 采用字典序方法比较新解z与殖民地λ,如果新解z优于殖民地λ,则新解z替代殖民地λ.

其中P_a为概率. 与GA的交叉不同,同化过程参与交叉的两个解中,殖民地是确定的,不是随机选择的.

3.3 革 命

殖民地革命模拟某种非预期的改变,例如,改变殖民地的语言或宗教,从而改变其特征和地位SBX[35]. 本文采用新的策略实现殖民地革命,具体过程描述如下.

Step 1: $l \leftarrow 1$,根据革命概率P_r确定殖民地数量,对于帝国k所有殖民地产生的[0, 1]之间的随机数u,若$u < P_r$,则$l \leftarrow l + 1$.

Step 2: 若$l > 1$,则执行如下过程:应用字典序方法选出成本最小的l个殖民地;对所选出的殖民地λ执行多项式变异,产生新解z;采用字典序方法比较新解z与殖民地λ,如果新解z优于殖民地λ,则新解z替代殖民地λ.

多项式变异是Deb等[37]提出的一种变异方法,也是多目标优化常用的变异算子,其计算公式为

$$y_{i,k} = x_{i,k} + \delta(x_{UB,k} - x_{LB,k}).$$

其中:$x_{UB,k}, x_{LB,k}$分别为第k个决策变量的上、下边界;δ根据下式确定:

$$\delta = \begin{cases} [2u + (1-2u)(1-\delta_1)^{\eta_m+1}]^{1/(\eta_m+1)}, \\ \quad u \leqslant 0.5; \\ 1 - [2(1-u) + 2(u-0.5)(1-\delta_2)^{\eta_m+1}]^{1/(\eta_m+1)}, \\ \quad u > 0.5. \end{cases} \tag{9}$$

η_m为用户选取的分布指数,一般实验中取11. 通常,参与革命的殖民地根据革命概率随机选择,本文首先根据革命概率确定参与革命的殖民地数量,然后针对优秀殖民地,执行多项式变异,这样利用优秀殖民地容易获得更好的殖民地,甚至产生新的殖民国家.

当同化和革命执行完毕后,进行互换操作,通过字典序方法判断殖民国家与殖民地之间的优劣从而可以判定是否有新的殖民国家出现. 若殖民地能替代原殖民国家,则该殖民地成为新的殖民国家,而原殖民国家成为殖民地.

3.4 殖民国家的差分进化

由于帝国内殖民地经常向殖民国家学习,如果殖民国家在较大代数内保持不变或者变化不大,则殖民地向殖民国家的学习将导致帝国内国家相似度增加,种群多样性降低,不利于算法逼近全局最优解; 另外,通常只有两种途径产生新殖民国家,即运用同化和革命产生的新殖民地来更替,这样不利于避免上述情况的出现. 为此,结合差分进化对殖民国家进行增强计

算,提供一种产生新殖民国家的新途径,以加快殖民国家的更新频率.

差分进化算法将一定比例的多个个体的差分信息作为个体的扰动量,使得算法在跳跃距离和搜索方向上具有自适应性. 在进化的早期,因为种群中个体的差异性较大使得扰动量较大,从而使得算法能够在较大范围内搜索,具有较强的探索能力; 而进化后期当算法趋向于收敛时,种群中个体的差异性较小,算法在个体附近搜索,这使得算法具有较强的局部开采能力[38]. 由于差分进化算法具有向种群个体学习的较强能力,使得其应用广泛.

殖民国家的差分进化过程描述如下:对于帝国k内的殖民国家z,运用差分进化产生新解z,采用字典序方法比较新解z与殖民国家,如果新解z优于殖民国家,则新解z更新殖民国家.

对解$x_i = (x_{i,1}, x_{i,2}, \cdots, x_{i,n})$,采用DE/rand/1公式,从种群$P$中随机确定$r_1 \neq r_2 \neq r_3 \neq i$,有

$$y_k = x_{r_1,k} + W \times (x_{r_2 k} - x_{x_3,k}); \tag{10}$$

$$z_k = \begin{cases} y_k, & u < \mathrm{CR} \text{ or } j = j_{\mathrm{rand}}; \\ x_{i,k}, & \text{Otherwise}. \end{cases} \tag{11}$$

其中:CR为交叉概率,W为缩放因子,j_{rand}为$[1, n]$内的随机整数,$z = (z_1, z_2, \cdots, z_n)$.

3.5　帝国竞争

帝国竞争是殖民地再分配的过程. 通常帝国竞争每代都进行. 由于过于频繁的帝国竞争会导致所有国家很快都聚集在一个帝国内,从而可能引起算法早熟. 本文采用不同于现有ICA文献[13-14,20,24]的策略,每隔I代进行一次帝国竞争,通过实验确定I取200. 当帝国竞争的条件成立后,执行如下帝国竞争过程.

Step 1: 计算每个帝国的势力$\mathrm{imper}_p(k), k = 1, 2, \cdots, N_{im}$.

Step 2: 计算每个帝国的概率q_k.

Step 3: 构建

$$D = [q_1 - u_1, q_2 - u_2, \cdots, q_{N_{im}} - u_{N_{im}}],$$

确定满足$q_k - u_k = \max\limits_i \{q_i - u_i\}$的帝国$k$,并将最差帝国的最差殖民地转移到帝国$k$中. 其中

$$u_i \sim U[0, 1],$$

$$q_k = \left| \mathrm{imper}_p(k) \Big/ \sum_{l=1}^{N_{im}} \mathrm{imper}_p(l) \right|.$$

重新定义帝国势力

$$\mathrm{imper}_p(k) = c_{\max} - c_k + \xi \sum_{g=1}^{NC_k} (c_{\max} - c_g). \tag{12}$$

其中:$\mathrm{imper}_p(k)$为第k个帝国的势力;$c_{\max}$为所有国家的最大的成本值;ξ为系数,取0.1.

直接利用上述公式计算概率q_k,这样省去了帝国归一化势力的计算,若采用基本ICA中的方法,则导致部分帝国归一化势力为0,这样这些帝国所对应的$q_i - u_i$将小于0,从而难以在帝国竞争中成为胜利者,会加快国家集聚在少量甚至一个帝国的速度.

3.6　算法描述

新型ICA的流程图如图2所示. 帝国消亡指帝国内无任何国家的情况,关于终止条件,采用目标函数估计次数,同时要记录算法进化的代数,即从同化到帝国消亡的各步骤循环的次数.

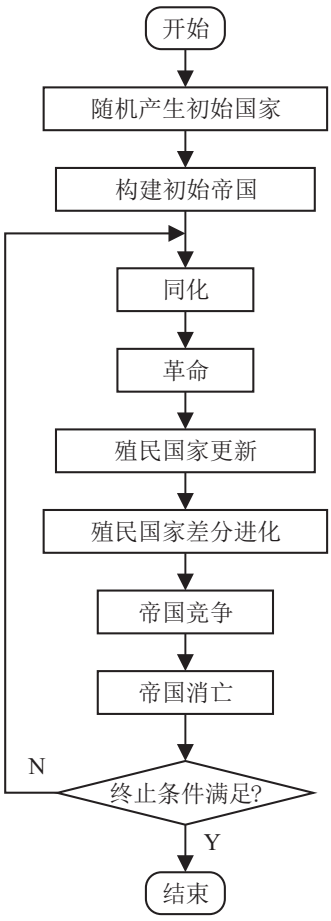

图 2　新型ICA流程

新型ICA的特点如下:对殖民国家的归一化成本和帝国的势力进行重新定义,以保证殖民国家和帝国势力都不为零,从而使所有殖民国家都能分配一定数量的殖民地,让所有帝国能参与竞争. 另外,在同化过程中引入了殖民地之间的全局搜索,以及殖民国家的差分进化等,这些策略增强了种群的多样性,能避免算法早熟.

4 实验分析与结果

为了验证新型ICA的性能,使用CEC2006关于COP[34]的12个测试函数和文献[39]的3个测试函数,其中前12个测试函数主要特征见表2. Fun表示目标函数的类型,LI表示线性不等式约束的个数,NI表示非线性不等式约束的个数,LE表示线性等式约束的个数,NE表示非线性等式约束的个数. $\rho = |\Omega|/|S|$ 表示可行域占整个搜索空间的比例,$|S|$ 表示从决策空间 S 中随机选出的个体,$|\Omega|$ 表示选出的 $|S|$ 个体中的可行解个数,一般有 $|S| = 1\,000\,000$. 文献[35]的第3节给出了后3个测试函数的详细描述.

表 2 12个测试函数

Problem	n	Fun	ρ/%	LI	NI	LE	NE
G01	13	quadratic	0.000 3	9	0	0	0
G02	20	nonlinear	99.997 3	1	1	0	0
G03	10	nonlinear	0.002 6	0	0	0	1
G04	5	quadratic	27.007 9	0	6	0	0
G05	4	nonlinear	0.000 0	2	0	0	3
G06	2	nonlinear	0.005 7	0	2	0	0
G07	10	quadratic	0.000 0	3	5	0	0
G08	2	nonlinear	0.858 1	0	2	0	0
G09	7	nonlinear	0.519 9	0	4	0	0
G10	8	linear	0.002 0	3	3	0	0
G11	2	quadratic	0.097 3	0	0	0	1
G12	3	quadratic	4.769 7	0	1	0	0

对于前12个测试函数,选用树种子算法(CTSA[40])和全局最好人工蜂群算法(MGABC[41])作为对比算法,CTSA是一种种群迭代搜索算法,其约束处理采用Deb's准则[42];MGABC是一种全局优化自然启发式优化算法,其约束处理采用一种新的惩罚函数法. CTSA中,ST $= 0.2$, Stand size $= 40$,终止条件为最大目标函数估计次数240 000,MGABC参数如文献[41]所示. 将后3个函数、新型ICA与文献[13,39]算法进行比较.

新型ICA的参数设置如下:种群规模 N 为100,殖民国家数量 $N_{im} = 6$,概率 $P_a = 0.7$,革命概率 $P_r = 0.1$,差分进化的交叉概率CR $= 0.9$,缩放因子 $W = 0.3$,每经过 I 代进行一次帝国竞争,通过实验确定 I 取200. 关于前12个测试函数,终止条件为最大目标函数估计次数200 000,关于后3个测试函数,最大目标函数估计次数为20 000. ICA关于每个测试函数分别独立运行20次,相应的计算结果如表3～表5所示. 其中:Optimal表示问题的目前最好解,Best和Mean表示最好解和平均解,Std.dev表示标准差,G03和G07的收敛曲线如图3和图4所示.

表 3 新型ICA的计算结果

函数	Optimal	Best	Mean	Std.dev
G01	−15.000	**−15.000**	−15.000	0
G02	−0.803 619	−0.791 906	0.700 005	0.050 786 3
G03	−1.000 5	−1.000 4	−1.000 19	6.2e-005
G04	−30 665.539	**−30 665.539**	−30 665.539	0
G05	5 126.498	5 126.498	5 126.81	0.382 026
G06	−6 961.814	**−6 961.814**	−6 961.814	0
G07	24.306	24.370 4	24.835 3	0.321 289
G08	0.095 825	**0.095 825**	0.095 825	0
G09	680.630	**679.429 404 140 248**	679.503	−0.045 061 8
G10	7 049.248	7 053.72	7 354.77	169.575
G11	0.749 90	**0.749 95**	0.749 95	0
G12	−1.000	**−1.000**	−1.000	0
Case 1	−8.116 46	**−8.116 46**	−8.116 46	7.479 47e-13
Case 2	−9.501 27	**−9.497 88**	−9.497 88	1.041 38e-09
Case 3	−7.506 09	−7.506 09	−7.193 89	0.349 046

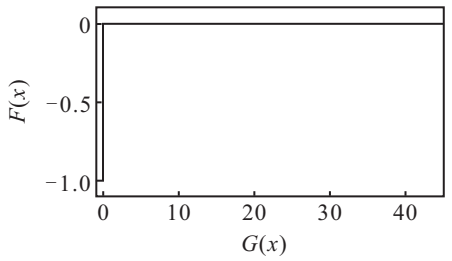

图 3 新型ICA关于G03收敛曲线

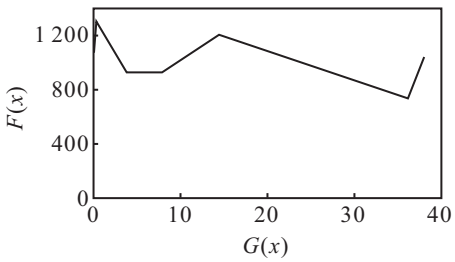

图 4 新型ICA关于G07收敛曲线

如表3所示,新型ICA关于7个函数获得了最优解,关于3个函数产生了与最优解非常接近的最好解,关于函数G09产生的结果优于现有的最好结果. 给出G09的最优解为

$$(2.329\,580\,435\,439\,305\,93,$$
$$1.937\,433\,271\,638\,394\,63,$$
$$0.535\,059\,694\,605\,449\,33,$$
$$4.397\,941\,491\,385\,374,$$

0.591 200 105 052 997 58,

0.794 858 964 394 850 93,

1.497 141 856 260 975 96),

目标函数为679.429 404 140 248, 这是近几年关于这组测试函数所获得的最新解. 除G02和G10之外的10个函数, 新型ICA的Mean接近Optimal, Std.dev也接近或达到相应的最好值. 另外, 如图3和图4所示, ICA的搜索过程中, 先逼近约束违背程度目标为0的那条直线或者纵轴, 然后逼近全局最优解, 这说明字典序方法是有效的. 如表5所示, ICA获得的最好解优于BIANCA[13], 关于Case1和Case3的结果优于文献[39]的算法, 而关于Case2的结果与rank-iMDDE[39]非常接近. 关于Case3, ICA给出了全新的最好解.

表4　3种算法关于12组测试函数的实验结果

函数	指标	新型ICA	MGABC	CTSA	函数	指标	新型ICA	MGABC	CTSA
	Best	−15.000 000	−15.000 000	—		Best	24.370 4	24.326 53	—
G01	Mean	−15.000 000	−13.553 54	−15.000 000	G07	Mean	24.835 3	24.780 64	24.551
	Std.dev	0	1.636 944	0		Std.dev	0.321 289	0.312 274 8	0.107
	Best	−0.791 906	−0.803 610 8	—		Best	0.095 825	0.095 825	—
G02	Mean	−0.700 005	−0.789 062 9	0.800 476	G08	Mean	0.095 825	0.095 825	0.095 825
	Std.dev	0.050 78	0.011 967 91	0.005 227		Std.dev	0	0	0
	Best	−1.000 4	−1.000 4	—		Best	679.429 404 140 248	680.630 2	—
G03	Mean	−1.000 19	−1.000 383	−1.000	G09	Mean	679.503	680.630 9	680.643
	Std.dev	6.200 5e-05	4.108 16e-05	0		Std.dev	0.045 061 8	0.000 512 664	0.006
	Best	−30 665.539	−306 65.54	—		Best	7 053.72	7 104.006	—
G04	Mean	−30 665.539	−30 665.54	−30 665.540	G10	Mean	7 354.77	7 357.461	7 139.377
	Std.dev	0	1.048 09e-11	0		Std.dev	169.575	121.903 1	61.068
	Best	5 126.498	5 126.497	—		Best	0.749 95	0.749 995	—
G05	Mean	5 126.812	5 467.756	5 223.609	G11	Mean	0.749 95	0.750 025	0.785
	Std.dev	0.382 026	330.868 1	131.111		Std.dev	0	0.000 034	0.039
	Best	−6 961.814	−6 961.803	—		Best	−1.000	−1.000 000	—
G06	Mean	−6 961.814	−6 959.489	−6 961.816	G12	Mean	−1.000	−1.000 000	−1.000
	Std.dev	0	1.176 998	0		Std.dev	0	0	0

表5　ICA和其他两种算法获得的最好解

函数	ICA(x_1, x_2)	BIANCA	rank-iMDDE
Case1	−8.116 46 (10.712 5, 2.963 38)	−8.099 33	−8.116 46
Case2	−9.497 88 (11.290 8, 2.468 39)	−9.497 83	−9.501 27
Case3	−7.506 09 (10.478 078 341 583 902 499 68, 2.734 005 465 343 691 376 75)	−7.376 96	−7.485 73

5　结　论

为求解COP, 本文提出了基于字典序的约束处理新策略, 并设计新型ICA对目标函数和约束违反程度同时优化. 该算法采用新策略定义国家的成本、归一化成本和帝国的势力, 在同化过程中引入殖民地之间的全局搜索, 选择殖民地中优秀个体参加革命, 运用差分进化对殖民国家增强计算, 并每隔一定代数进行帝国竞争以防止算法早熟. 通过对测试函数的求解以及与对比算法的性能比较, 表明了新型ICA在求解COP方面具有较强的竞争力.

参考文献(References)

[1] 李智勇, 黄滔, 陈少淼, 等. 约束优化进化算法综述[J]. 软件学报, 2017, 28(6): 1529-1546.
(Li Z Y, Huang T, Chen S M, et al. Overview of constrained optimization evolutionary algorithms[J]. J of Software, 2017, 28(6): 1529-1546.)

[2] 王勇, 蔡自兴, 周育人. 约束优化进化算法[J]. 软件学报, 2009, 20(1): 11-29.
(Wang Y, Cai Z X, Zhou Y R. Constrained optimization evolutionary algorithm[J]. J of Software, 2009, 20(1): 11-29.)

[3] Mezura-Montes E, Coello C A C. Constraint-handling in nature-inspired numerical optimization: Past, present and future[J]. Swarm and Evolutionary Computation, 2011, 1(4): 173-194.

[4] Gong W, Cai Z H, Liang D W. Adaptive ranking mutation operator based differential evolution for constrained optimization[J]. IEEE Trans on Cybernetics, 2015, 45(4): 716-727.

[5] Saha A, Datta R, Deb K. Hybrid gradient projection based genetic algorithms for constrained optimization[C]. Proc of the Congress on Evolutionary Computation. Barcelona: IEEE Service Center, 2010: 2851-2858.

[6] Runarsson T P, Yao X. Stochastic ranking for constrained evolutionary optimization[J]. IEEE Trans on Evolutionary Computation, 2000, 4(3): 284-294.

[7] Takahama T, Sakai S. Constrained optimization by the ε constrained differential evolution with gradient-based mutation and feasible elites[C]. Proc of the 2006 IEEE Int Conf on Evolutionary Computation. Vancouver: IEEE Press, 2006: 372-378.

[8] Wang Y, Cai Z X, Guo G Q, et al. Multi-objective optimization and hybrid evolutionary algorithm to solve constrained optimization problems[J]. IEEE Trans on Systems, Man, and Cybernetics, 2007, 37(3): 560-575.

[9] Peng S Y. Hybrid immune clonal particle swarm optimization multi-objective algorithm for constrained optimization problems[J]. Int J of Patter Recognition and Artificial Intelligence, 2017, 31(1): 2495-2508.

[10] Mallipeddi R, Suganthan P N. Ensemble of constraint handling techniques[J]. IEEE Trans on Evolutionary Computation, 2010, 14(4): 561-579.

[11] 甘敏, 彭辉, 王勇. 多目标优化与自适应惩罚的混合约束优化进化算法[J]. 控制与决策, 2010, 25(3): 378-382.
(Gan M, Peng H, Wang Y. Multiobjective optimization and adaptive penalty function based constrained optimization evolutionary algorithm[J]. Control and Decision, 2010, 25(3): 378-382.)

[12] 毕晓君, 张磊. 基于混合策略的双种群约束优化算法[J]. 控制与决策, 2015, 30(4): 715-720.
(Bi X J, Zhang L. Dual population constrained optimization algorithm with hybird strategy[J]. Control and Decision, 2015, 30(4): 715-720.)

[13] Gong W, Cai Z, Liang D. Engineering optimization by means of an improved constrained differential evolution[J]. Computer Methods in Applied Mechanics and Engineering, 2014, 268: 884-904.

[14] Gao W F, Yen G, Liu S Y. A dual-population differential evolution with coevolution for constrained optimization[J]. IEEE Trans on Cybernetics, 2014, 45(5): 1108-1121.

[15] He Q, Wang L. An effective co-evolutionary particle swarm optimization for constrained engineering design problems[J]. Engineering Applications of Artificial Intelligence, 2007, 20(1): 89-99.

[16] Wang L , Fang C. An effective estimation of distribution algorithm for the multi-mode resource-constrained project scheduling problem[J]. Computers & Operations Research, 2012, 39(2): 449-460.

[17] Fang C, Wang L, Xu Y. An estimation of distribution algorithm for resource-constrained project scheduling problem[C]. Proc of 2010 Chinese Control and Decision Conf. Xuzhou: IEEE, 2010: 265-270.

[18] He Q, Wang L. A hybrid particle swarm optimization with a feasibility-based rule for constrained optimization[J]. Applied Mathematics & Computation, 2007, 186(2): 1407-1422.

[19] Enayatifar R, Yousefi M, Abdullah A H, et al. MOICA: A novel multi-objective approach based on imperialist competitive algorithm[J]. Applied Mathematics and Computation, 2013, 219(17): 8829-8841.

[20] Bilel N, Mohamed N, Zouhaier A, et al. An improved imperialist competitive algorithm for multi-objective optimization[J]. Engineering Optimization, 2016, 48(11): 1823-1844.

[21] Lian K, Zhang C, Gao L, et al. Single row facility lay-out problem using an imperialist competitive algorithm[C]. Proc of the 41st Int Conf on Computers and Industrial Engineering. Los Angeles: Elsevier, 2011: 578-586.

[22] Hosseini S, Khaled A A, Vadlamani S. Hybrid imperialist competitive algorithm, variable neighborhood search, simulated annealing for dynamic facility layout problem[J]. Neural Computing and Applications, 2014, 25(7): 1871-1885.

[23] Karimi N, Zandieh M, Najafl A A. Group scheduling in flexible flow shops: A hybridised approach of imperialist competitive algorithm and electromagnetic-like mechanism[J]. Int J of Production Research, 2011, 49(16): 4965-4977.

[24] Behnamian J, Zandieh M. A discrete colonial competitive algorithm for hybrid flow shop scheduling to minimize earliness and quadratic tardiness penalties[J]. Expert Systems with Applications, 2011, 38(12): 14490-14498.

[25] Goldansaz S M, Jolai F, Anaraki A H Z. A hybrid imperialist competitive algorithm for minimizing makespan in a multi-processor open shop[J]. Applied Mathematical Modelling. 2013, 37(23): 9603-9616.

[26] Naderi B, Yazdani M. A model and imperialist competitive algorithm for hybrid flow shops with sublots and setuptimes[J]. J of Manufacturing Systems, 2014,

33(4): 647-653.

[27] Karimi S, Ardalan Z, Naderi B, et al. Scheduling flexible job-shops with transportation times: Mathematical models and a hybrid imperialist competitive algorithm[J]. Applied Mathematical Modelling, 2017, 41(1): 667-682.

[28] 雷德明, 杨冬婧. 具有总能耗约束的柔性作业车间调度问题研究[J]. 自动化学报, 2018. 44(11): 2083-2091.
(Lei D M, Yang D J. Research on flexible job shop scheduling problem with total energy consumption constraint [J]. Acta Automation Sinica, 2018, 44(11): 2083-2091.)

[29] Zhou W, Yan J, Li Y, et al. Imperialist competitive algorithm for assembly sequence planning[J]. Int J of Advanced Manufacturing Technology, 2013, 67(9): 2207-2216.

[30] Wang B, Guan Z, Li D, et al. Two-sided assembly line balancing problems with operator number and task constraints: A hybrid imperialist competitive algorithm[J]. Int J of Advanced Manufacturing Technology, 2014, 74(5): 791-805.

[31] 张鑫龙, 郑秀万, 肖汉, 等. 一种求解旅行商问题的新型帝国竞争算法[J]. 控制与决策, 2016, 31(4): 586-592.
(Zhang X L, Zheng X W, Xiao H, et al. A new imperialist competitive algorithm for solving TSP problem[J]. Control and Decision, 2016, 31(4): 586-592.)

[32] Hosseini S, Khaled A A. A survery on the imperialist competitive algorithm metaheuristic: Implementation in engineering domain and directions for future research[J]. Applied Soft Computing, 2014, 24(C): 1078-1094.

[33] Wang Y, Cai Z X. Combining multi-objective optimization with differential evolution to solve constrained optimization problems[J]. IEEE Trans on Evolutionary Computation, 2012, 16(1): 117-134.

[34] Liang J, Runarsson T P, Mezura-Montes E. Problem definitions and evaluation criteria for the CEC 2006 special session on constrained real-parameter optimization[R]. Singapore: Nanyang Technological University, 2006.

[35] Atashpaz-Gargari E, Lucas C. Imperialist competitive algorithm: An algorithm for optimization inspired by imperialistic competition[C]. Proc of the 2007 IEEE Congress on Evolutionary Computation (CEC'07). Singapore, 2007: 4661-4667.

[36] Deb K, Agrawal R B. Simulated binary crossover for continuous search space[J]. Complex Systems, 1995, 9(2): 115–148.

[37] Deb K, Goyal M. A combined genetic adaptive search (Gene AS) for engineering design[J]. Computer Science and Informatics, 1996, 26(4): 30-45.

[38] Price K, Storn R, Lampinen J. Differential evolution: A practical approach to global optimization[M]. New York: Springer-Verlag, 2005.

[39] Montemurro M, Vincenti A, Vannucci P. The automatic dynamic penalisation method (ADP) for handling constraints with genetic algorithms[J]. Computer Methods in Applied Mechanics and Engineering, 2013, 256(4): 70-87.

[40] Babalik A, Cinar C A, Kiran S M. A modification of tree-seed algorithm using Deb's rules for constrained optimization[J]. Applied Soft Computing, 2018, 63(2): 289-305.

[41] Bansal C J, Joshi K S, Sharma H. Modified global best artificial bee colony for constrained optimization problems[J]. Computers and Electrical Engineering, 2018, 67(4): 365-382.

[42] Deb K. An efficient constraint handling method for genetic algorithms[J]. Computer Methods in Applied Mechanics and Engineering, 2000, 186(2/3/4): 311-338.

作者简介

雷德明(1968–), 男, 教授, 博士, 从事智能系统优化与控制等研究, E-mail: demingleill@163.com;

操三强(1993–), 男, 硕士生, 从事智能计算的研究, E-mail: caosanqiang93@163.com;

李明(1986–), 男, 博士生, 从事制造系统智能优化与调度的研究, E-mail: 1039518062@qq.com.

(责任编辑: 郑晓蕾)