控制与决策
Control and Decision

基于改进的Tent混沌万有引力搜索算法

张娜， 赵泽丹， 包晓安， 钱俊彦， 吴彪

引用本文:
张娜, 赵泽丹, 包晓安, 等. 基于改进的Tent混沌万有引力搜索算法[J]. 控制与决策, 2020, 35(4): 893−900.

在线阅读 View online: https://doi.org/10.13195/j.kzyjc.2018.0795

文章编号: 1001-0920(2020)04-0893-08 DOI: 10.13195/j.kzyjc.2018.0795

基于改进的Tent混沌万有引力搜索算法

张 娜[1], 赵泽丹[1], 包晓安[1†], 钱俊彦[2], 吴 彪[1]

(1. 浙江理工大学 信息学院，杭州 310018；2. 桂林电子科技大学 广西可信软件重点实验室，广西 桂林 541004)

摘 要: 万有引力搜索算法(gravitational search algorithm, GSA)相比于传统的优化算法具有收敛速度快、开拓性能强等特点,但GSA易陷入早熟收敛和局部最优,搜索能力较弱. 为此,提出一种基于改进的Tent混沌万有引力搜索算法(gravitational search algorithm based on improved tent chaos, ITC-GSA). 首先,改进Tent混沌映射来初始化种群,利用Tent混沌序列随机性、遍历性和规律性的特性使得初始种群随机性和遍历性在可行域内,具有加强算法的全局搜索能力;其次,引入引力常数G的动态调整策略提高算法的收敛速度和收敛精度;再次,设计成熟度指标判断种群成熟度,并使用Tent混沌搜索有效抑制算法早熟收敛,帮助种群跳出局部最优;最后,对10个基准函数进行仿真实验,结果表明所提算法能够有效克服GSA易陷入早熟收敛和局部最优的缺点,提高算法的收敛速度和寻优精度.

关键词: Tent混沌；万有引力搜索算法(GSA)；成熟度；引力常数

中图分类号: TP273 **文献标志码:** A

Gravitational search algorithm based on improved Tent chaos

ZHANG Na[1], *ZHAO Ze-dan*[1], *BAO Xiao-an*[1†], *QIAN Jun-yan*[2], *WU Biao*[1]

(1. School of Informatics and Electronics，Zhejiang Sci-Tech University，Hangzhou 310018，China；2. Guangxi Key Laboratory of Trusted Software，Guilin University of Electronic Technology，Guilin 541004，China)

Abstract: The gravitational search algorithm (GSA) has the characteristics of faster convergence speed and stronger exploitation performance than the traditional optimization algorithm, but the GSA is vulnerable to premature convergence and local optimum, and its search ability is weak. Therefore, this paper proposes the gravitational search algorithm based on improved Tent chaos (ITC-GSA). Firstly, the Tent chaotic map is improved to initialize the population. Using the characteristics of randomness, ergodicity and regularity of Tent chaotic sequence, the initial population randomness and ergodicity are within the feasible domain, and the global search ability of the algorithm is enhanced. Then, the dynamic adjustment strategy of the gravity constant G is introduced to improve the convergence speed and convergence accuracy of the algorithm. Moreover, maturity indicators are designed to determine the maturity of the population, and Tent chaos search is used to effectively suppress the premature convergence of the algorithm for helping the population jump out of the local optimum. Finally, through simulations of 10 benchmark functions, experiments show that the proposed algorithm can effectively overcome the shortcomings of the GSA's vulnerability to premature convergence and local optimization, and improve the algorithm's convergence speed and optimization accuracy.

Keywords: Tent chaos；gravitational search algorithm；maturity；gravitational constant

0 引 言

优化理论的应用领域十分广泛,因此众多学者投入到优化理论的研究中. 其中的启发式优化算法备受关注,并在近年来得到长足的发展,而由伊朗克曼大学教授Rashedi等[1]在2009年提出的万有引力搜索算法(gravitational search algorithm, GSA)是启发式优化算法的一个典型代表.

GSA算法是受到牛顿万有引力的启发而提出的启发式算法,其让种群粒子具有质量,且质量根据粒子的适应度值计算出. 同时,基于牛顿万有引力定律,粒子间产生相互作用力,作用力根据改进的万有引力公式得出,粒子间的作用力与它们的质量成正比,与距离的平方成反比[2]. 粒子质量越大,粒子间距离越小,则粒子间的相互作用力越大. 粒子在作用力的驱

收稿日期: 2018-06-10；**修回日期:** 2018-08-02.

基金项目: 国家自然科学基金项目(61502430, 61562015)；广西自然科学重点基金项目(2015GXNSFDA139038)；浙江理工大学521人才培养计划项目.

责任编委: 陈家伟.

†通讯作者. E-mail: baoxiaoan@zstu.edu.cn.

动下作相对运动. 适应度值大的粒子则质量较大,适应度值小的粒子则质量较小. 在相同的作用力下,质量大的相对运动较慢,质量小的则运动较快. 粒子逐渐收敛到最优位置,达到寻优的目的.

GSA算法相较于其他智能优化算法具有结构简单,参数较少,全局寻优能强等特点. 同时,GSA也存在着易陷入早熟收敛和局部最优、搜索能力弱等缺点. 针对GSA存在的缺点,很多学者提出了改进: 文献[3]在万有引力搜索算法中引入混沌变异,在一定程度上避免了早熟问题; 文献[4]运用混沌策略进行局部搜索,提高了算法的开拓性,提升了算法的寻优精度; 文献[5]将混沌引入GSA算法中,首先引入了一个具有无限折叠的迭代混沌映射,并设计了混沌局部搜索,然后将混沌局部搜索和基本万有引力搜索合并到寻优过程框架中; 文献[6]将猫混沌映射引入到GSA中,在初始化时使用猫混沌初始化种群,在陷入早熟时引入一个小的混沌干扰来跳出早熟; 文献[7]将Logistic映射引入到GSA中,使用Logistic映射生成的混沌序列替换随机序列,然后使用混沌作为局部搜索方法.

上述文献中的改进虽然能在一定程度上避免算法陷入局部最优,提高搜索能力,但是在收敛速度和寻优精度上仍存在不足,且虽然引入混沌策略,但只是将两者结合起来,并未对混沌策略和GSA算法本身做出改进. 本文提出基于改进的Tent混沌万有引力搜索算法(gravitational search algorithm based on improved tent chaos, ITC-GSA),该算法首先使用Tent混沌映射初始化种群,然后使用动态调整的引力常数G进行迭代搜索,同时引入成熟度指标判断种群成熟度,当种群出现早熟或陷入局部最优时,使用Tent混沌搜索帮助粒子跳出早熟或局部最优. 本文对10个测试函数进行仿真实验,结果表明,相比于GSA, ITC-GSA无论在收敛速度还是在寻优精度上都有极大的提升.

1 Tent混沌序列

Tent混沌序列具有随机性、遍历性和规律性的特征[8],利用这些特征进行优化搜索,能够有效地保持种群多样性,抑制算法落入局部最优,使全局搜索能力得到改善. 研究表明,不同的混沌映射对混沌优化过程的影响很大[9]. 已经有众多学者将混沌映射和混沌搜索思想引入到万有引力搜索算法中,但是从目前的文献来看,在优化启发式算法中运用相对较多的是Logistic混沌映射. 单梁等[10]的研究表明,Tent混沌映射具有比Logistic混沌映射更好的遍历均匀性和更快的搜索速度. 图1和图2分别为Tent混沌序列的直方图和分布图,图3和图4分别为Logistic混沌序列的直方图和分布图,通过这4个图可以发现Logistic混沌映射在$[0, 0.05]$和$[0.95, 1]$范围内的取值概率高于其他各段的取值概率,而Tent在可行域各段的取值概率则相对均匀. Tent混沌映射产生的Tent混沌序列的均匀性明显优于Logistic混沌映射产生的Logistic混沌序列. 研究表明,Logistic混沌映射的不均匀性对寻优速度和寻优精度影响较大[9],将Logistic混沌映射应用于初始化种群时,其混沌序列的不均匀性会导致初始化种群分布的不均匀,影响算法在寻优过程中的速率和精度. 本文充分利用Tent的遍历性产生均匀分布的混沌序列,减少初始值对算法优化的影响.

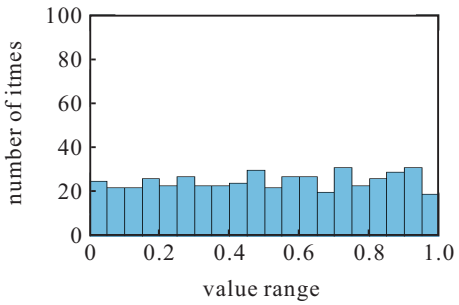

图1 Tent混沌序列分布直方图

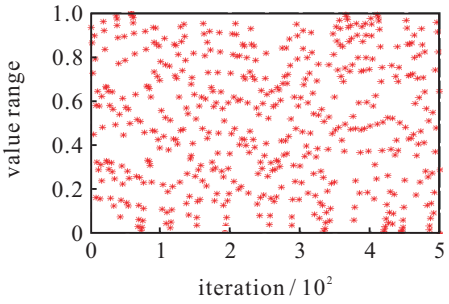

图2 Tent混沌序列分布图

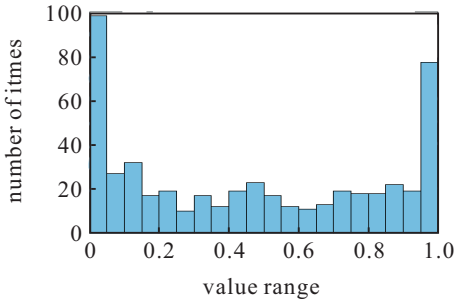

图3 Logistic混沌序列分布直方图

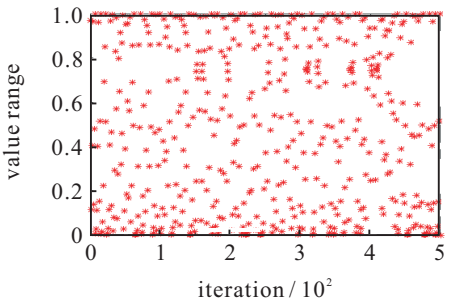

图4 Logistic混沌序列分布图

Tent混沌映射的表达式如下:

$$x_{i+1}=\begin{cases}2x_i,\ 0\leqslant x\leqslant\dfrac{1}{2};\\ 2(1-x_i),\ \dfrac{1}{2}<x\leqslant 1.\end{cases}\tag{1}$$

Tent混沌映射通过贝努利移位变换后表示如下:

$$x_{i+1}=(2x_i)\bmod 1.\tag{2}$$

分析Tent混沌迭代序列能够发现序列中存在小周期,并且存在不稳周期点. 为避免Tent混沌序列在迭代时落入小周期点和不稳定周期点,在原有的Tent混沌映射表达式上引入一个随机变量$\mathrm{rand}(0,1)\times\dfrac{1}{N}$,则改进后的Tent混沌映射表达式如下:

$$x_{i+1}=\begin{cases}2x_i+\mathrm{rand}(0,1)\times\dfrac{1}{N},\ 0\leqslant x\leqslant\dfrac{1}{2};\\ 2(1-x_i)+\mathrm{rand}(0,1)\times\dfrac{1}{N},\ \dfrac{1}{2}<x\leqslant 1.\end{cases}\tag{3}$$

变换后的表达式为

$$x_{i+1}=(2x_i)\mathrm{mod}1+\mathrm{rand}(0,1)\times\frac{1}{N}.\tag{4}$$

其中: N 是序列内粒子的个数,$\mathrm{rand}(0,1)$ 是范围在$[0,1]$之间的随机数. 引入随机变量$\mathrm{rand}(0,1)\times\dfrac{1}{N}$不仅仍然保持了Tent混沌映射的随机性、遍历性、规律性,而且能够有效避免迭代落入小周期点和不稳定周期点内. 本文算法引入的随机变量,既保持了随机性,又将随机值控制在一定的范围之内,保证了Tent混沌的规律性. 根据Tent混沌映射的特性,本文利用改进的Tent混沌映射表达式在可行域中产生Tent混沌序列,其步骤如下.

step 1: 在区间$(0,1)$内取随机初值x_0,即$i=0$;

step 2: 根据式(3)进行迭代计算,产生一个X序列,每次迭代i自增1;

step 3: 当i等于最大迭代次数时停止迭代,保存产生的X序列.

2　万有引力搜索算法

2.1　万有引力算法

GSA是模拟物理学上万有引力定律进行搜索的启发式优化算法. 在GSA中,将优化问题的解视为一组在空间运行的粒子[11],假设在一个D维搜索空间中,存在N个粒子,则第i个粒子在D维搜索空间中的位置为$X_i=[x_i^1,\ldots,x_i^d,\ldots,x_i^D]$,其中$i=1,2,\ldots,N$,$x_i^d$表示第$i$个粒子在第$d$维的位置. Rashedi等[1]在万有引力定律的启发下给出粒子在搜索空间中的受力公式,定义在t时刻第d维空间上粒子i受到粒子j的作用力为$F_{ij}^d(t)$,表达式如下:

$$F_{ij}^d(t)=G(t)\frac{M_i(t)\times M_j(t)}{R_{ij}(t)+\varepsilon}(x_j^d(t)-x_i^d(t)).\tag{5}$$

其中: $M_i(t)$和$M_j(t)$分别是t时刻粒子i和j的惯性质量;$G(t)$是t时刻的引力系数;ε是一个很小的常量;$R_{ij}(t)$是t时刻粒子i和j之间的欧氏距离,表示为

$$R_{ij}(t)=\|X_i(t),X_j(t)\|_2.\tag{6}$$

为了增加随机性,定义在d维空间上粒子i受到的合力为$F_i^d(t)$,表达式为

$$F_i^d(t)=\sum_{j=1,j\neq i}^{N}\mathrm{rand}(0,1)_jF_{ij}^d(t),\tag{7}$$

其中$F_{ij}^d(t)$是t时刻在d维空间上粒子i和j之间的作用力.

根据牛顿第二定律,定义t时刻第d维空间上粒子i的加速度为$a_i^d(t)$,表达式为

$$a_i^d(t)=\frac{F_i^d(t)}{M_i(t)}.\tag{8}$$

利用适应度值,根据式(9)和(10)更新粒子i的惯性质量$M_i(t)$,有

$$m_i(t)=\frac{\mathrm{fit}_i(t)-\mathrm{worst}(t)}{\mathrm{best}(t)-\mathrm{worst}(t)},\tag{9}$$

$$M_i(t)=\frac{m_i(t)}{\sum_{j=1}^{N}m_j(t)}.\tag{10}$$

对于求取最小值问题,$\mathrm{best}(t)$和$\mathrm{worst}(t)$的定义为

$$\mathrm{best}(t)=\min_{i\in\{1,2,\ldots,N\}}\mathrm{fit}_i(t),\tag{11}$$

$$\mathrm{worst}(t)=\max_{i\in\{1,2,\ldots,N\}}\mathrm{fit}_i(t);\tag{12}$$

对于求取最大值问题,$\mathrm{best}(t)$和$\mathrm{wrost}(t)$的定义为

$$\mathrm{best}(t)=\max_{i\in\{1,2,\ldots,N\}}\mathrm{fit}_i(t),\tag{13}$$

$$\mathrm{worst}(t)=\min_{i\in\{1,2,\ldots,N\}}\mathrm{fit}_i(t),\tag{14}$$

其中$\mathrm{fit}_i(t)$是粒子i在t时刻的适应度值. 在GSA中,每次迭代运算,粒子i都根据以下公式进行速度和位置的更新:

$$v_i^d(t+1)=\mathrm{rand}(0,1)_i\times v_i^d(t)+a_i^d(t),\tag{15}$$

$$x_i^d(t+1)=x_i^d(t)+v_i^d(t+1).\tag{16}$$

分析GSA可以发现,粒子间引力的作用使得惯性质量小的粒子移动的步长较长,便于搜索全局最优,体现了GSA的搜索性能;惯性质量大的粒子移动的步长较小,便于保持和开拓当前的最优解,体现了GSA的开拓性能. 但文献[12-13]指出,GSA算法收敛速度较其他优化算法速度较快,容易早熟收敛和陷入局部最优.

2.2 引力常数G的动态调整策略

GSA在解决特定问题时,参数的控制是至关重要的,即引力常数G、种群规模N、迭代次数T和衰减速率的量α[14]. 引力常数G直接影响着粒子受到的合力大小以及加速度,引力常数G对GSA中粒子能否跳出局部最优和提高最优解精度都有重要的影响,对引力常数的合理设置可保证GSA的搜索性能[15]. 引力常数G的表达式为

$$G(t)=G_0\mathrm{e}^{-\alpha\frac{t}{T}}. \tag{17}$$

其中:G_0为引力常数的初始值,α为描述衰减速率的量,T为最大迭代的次数,t为当前迭代的次数. 由文献[1]可知,当$G_0=100,\alpha=20$时,GSA的性能最优.

引力常数$G(t)$的大小在迭代初期随着t的增加而快速下降,当t足够大时趋于平稳,即$G(t)$在迭代早期下降较快,在迭代末期下降较慢. 引力常量G对粒子移动的步长具有重要影响,相对情况下,G越大则步长越长,反之则步长越短. 本文引入引力常数$G(t)$动态调整策略设计引力常数,有

$$G'(t)=G_0\mathrm{e}^{-\alpha\frac{t}{T}\times(\omega_1+\omega_2\times\frac{t}{T})}, \tag{18}$$

其中ω_1、ω_2为权值(根据经验,取$\omega_1=1.5,\omega_2=1$). 如图5所示,$G'(t)<G(t),G(t)$在迭代初期采取大步移动,在迭代末期则缓慢移动[16]. 迭代早期,$G'(t)$有利于控制步长,防止粒子跳过适应度值高的点,增加粒子多样性的同时保持良好的搜索能力;迭代末期,有利于粒子精细化寻优,增强粒子的开拓性能. 引力常数$G(t)$动态调整策略保证了粒子收敛速度的动态调整,有效地避免了粒子早熟收敛和陷入局部最优,提高了寻优精度.

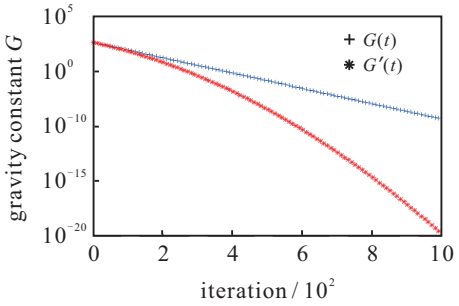

图5 引力常数$G(t)$和$G'(t)$与迭代次数t的关系

3 成熟度指标设计及Tent混沌搜索

3.1 成熟度指标

成熟度表示种群中粒子间的相似程度,粒子间相似程度越高,则种群成熟度越高. 成熟度指标用于早熟收敛和陷入局部最优的判断,当种群未满足收敛准则,而成熟度却较高的,说明可能已经早熟或陷入局部最优.

由于粒子间相似程度具有模糊性,模糊理论的成熟度指标可以全面并且准确地反映种群成熟度的情况,因此本文选取模糊指标作为成熟度的评价指标. 用种群平均贴近度作为成熟度指标,种群平均贴近度越大,则粒子间相似性越大,成熟度越高. 当种群平均贴近度大于阀值时,说明种群聚集到一起,该区域的粒子密度较高,表明种群平均贴近度较高,种群可能陷入早熟或局部最优.

设在搜索空间D中,种群规模为N,第t代第i个粒子的位置为$X_i(t)=(x_i^1(t),x_i^2(t),\dots,x_i^D(t))$,第$i$个粒子个体迭代过程中经历的最优位置为$\mathrm{pbest}_i(t)=(\mathrm{pbest}_i^1(t),\mathrm{pbest}_i^2(t),\dots,\mathrm{pbest}_i^D(t))$,将所有粒子现在位置与迭代过程中的最优位置组成一个$N\times 2D$阶矩阵$Q(t)$,即

$$Q(t)=\begin{bmatrix} x_1^1(t) & x_1^2(t) & \dots & x_1^D(t) & \mathrm{pbest}_1^1(t) & \mathrm{pbest}_1^2(t) & \dots & \mathrm{pbest}_1^D(t)\\ x_2^1(t) & x_2^2(t) & \dots & x_2^D(t) & \mathrm{pbest}_2^1(t) & \mathrm{pbest}_2^2(t) & \dots & \mathrm{pbest}_2^D(t)\\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots\\ x_N^1(t) & x_N^2(t) & \dots & x_N^D(t) & \mathrm{pbest}_N^1(t) & \mathrm{pbest}_N^2(t) & \dots & \mathrm{pbest}_N^D(t) \end{bmatrix}. \tag{19}$$

对式(19)作归一化运算,可以得到$N\times 2D$阶矩阵$Q'(t)$,其中

$$Q'(t)_{uv}=\frac{Q_{uv}(t)-\min\limits_{1\leqslant n\leqslant N,1\leqslant d\leqslant 2D}Q_{nd}(t)}{\max\limits_{1\leqslant n\leqslant N,1\leqslant d\leqslant 2D}Q_{nd}(t)-\min\limits_{1\leqslant n\leqslant N,1\leqslant d\leqslant 2D}Q_{nd}(t)}. \tag{20}$$

矩阵$Q'(t)$中每个行向量$Q'_i(t)$表示一个模糊集合,$Q'(t)$中任意两个模糊集合$Q'_i(t)$与$Q'_j(t)$的相似度用贴近度表示,记为

$$\sigma(i,j)=1-\frac{1}{2D}\sum_{v=1}^{2D}|q'_{iv}(t)-q'_{jv}(t)|, \tag{21}$$

其中$q'_{iv}(t)$为$Q'_i(t)$的第v个元素. 用种群平均贴近度表示种群成熟度,记为

$$S=\frac{2\sum\limits_{i=1}^{N-1}\sum\limits_{j=i+1}^{N}\sigma(i,j)}{N(N-1)}. \tag{22}$$

种群平均贴近度为$0\leqslant S\leqslant 1$,S越大表明种群成熟度越高.利用种群平均贴近度可以有效判断种群是否早熟或陷入局部最优,如果当前迭代次数远小于最大

迭代次数,但此时种群平均贴近度较高时,说明种群有可能早熟或陷入局部最优.

3.2 Tent混沌搜索

本文引入Tent混沌搜索,有利于算法跳出局部最优,从而提高寻优的精度. Tent混沌搜索是将种群X映射到$(0,1)$区间,通过Tent混沌映射进行迭代,产生新的混沌序列,再将新的混沌序列载波到原搜索空间,得到一个新的种群Y,最后将新种群与原种群进行比较,保留优质的粒子,将优质的粒子组成一个种群Z.

Tent混沌搜索空间参数描述如下: $[X^d_{\min}, X^d_{\max}]$为搜索空间,其中$X^d_{\min}$为第d维向量的最小值,$X^d_{\max}$为最大值. 搜索停滞的解为$X_k = (x^1_k, x^2_k, \cdots, x^D_k)$. 混沌搜索的步骤如下.

step 1: 利用

$$Z^d_k(0) = \frac{x^d_k - X^j_{\min}}{X^j_{\max} - X^j_{\min}} \tag{23}$$

将X_k映射到$(0,1)$. 其中: $k = 1,2,\ldots,n$表示第n个粒子, $d = 1,2,\ldots,D$表示维度.

step 2: 将$Z^d_k(0)$代入式(3)的Tent混沌映射进行迭代运算,产生一个混沌序列$Z^d_k(t)(t = 1,2,\ldots, C_{\max})$,其中$C_{\max}$表示混沌搜索的最大迭代次数.

step 3: 利用

$$W^d_k = x^d_k + \frac{x^d_{\max} - X^d_{\min}}{2}(2Z^d_k(t) - 1) \tag{24}$$

将$Z^d_k(t)$载波到原搜索空间,得到的解为W_k.

step 4: 计算W_k的适应度$F(W_k)$,比较W_k的适应度$F(W_k)$和搜索停滞解X_k的适应度$F(X_k)$,取适应度值高的解作为生成的新解.

step 5: 达到最大混沌搜索次数,则混沌搜索结束,否则转到step 2.

当粒子陷入早熟收敛和局部最优时,搜索停滞的解为$X_k = (x^1_k, x^2_k, \ldots, x^D_k)$,根据Tent混沌搜索生成新的解,新解的适应度值高于搜索停滞解的适应度,达到跳出早熟收敛和局部最优的目的.

4 Tent混沌的万有引力搜索算法

改进的Tent混沌万有引力搜索算法(ITC-GSA)的搜索过程可以归纳为3大部分. 第1部分改进Tent混沌序列产生初始种群,生成的初始化种群具有遍历性和随机性,能够有效避免早熟收敛和陷入局部最优. 第2部分在搜索阶段使用引力常数$G(t)$的动态调整策略. 第3部分引入成熟度指标判断种群成熟度,并引入Tent混沌搜索使种群能够有效跳出局部最优. 当种群平均贴近度S较高(为有效判断种群早熟,同时保持种群粒子的多样性,根据经验,本文取$0.9 \leqslant S \leqslant 1$,即阀值取0.9),且当前迭代次数$t$小于最大迭代次数$T$时,使用Tent混沌搜索得出新解,更新粒子位置(如果每一次迭代都计算种群贴近度S,则大大影响了算法的速率,为提高算法的效率可设置迭代n代后间隔m代计算种群平均贴近度).

随机初始化种群粒子的不均匀性对算法的寻优速度影响较大[9],在初始化种群时,ITC-GSA使用改进的Tent混沌序列产生初始化种群,相较于GSA算法使用的随机初始化种群不仅提高了收敛速度而且利于后续寻优精度的提高. 同时,引力常数$G(t)$的动态调整在搜索前期有利于粒子的全局搜索,具有良好的搜索能力,在搜索末期则有利于增强开拓能力,提高寻优精度. 同时使用成熟度指标判断种群是否早熟收敛或者陷于局部最优,利用Tent混沌搜索使得粒子避免早熟或跳出局部最优. 算法流程如图6所示.

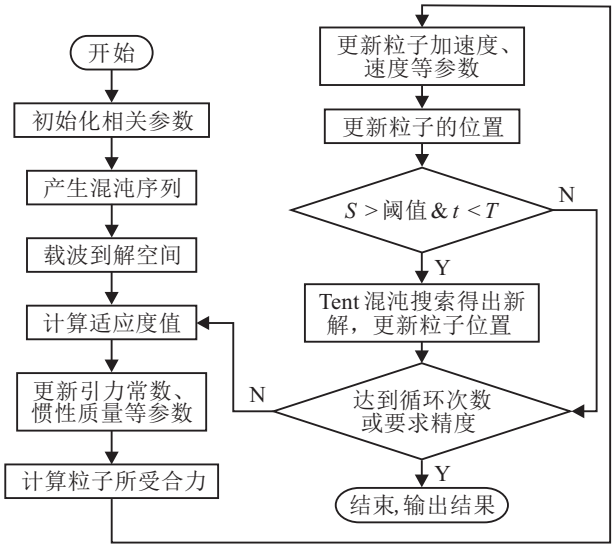

图6　算法流程

以下是ITC-GSA的伪代码,其中T为最大迭代次数, AC为精度, f_{best}为最优值, $\text{fun}(x)$为x的函数值, g_{best}为最优解.

1) initialize related parameters

2) randomly generate vector $Z^d_k(0)$ in D-dimensional space

3) generate N non-equal chaotic sequences $Z^d_k(t)$ according to Eq.(3)

4) get initialized individual X_i according to Eq.(6)

5) $g_{\text{best}} = X_i$ and $f_{\text{best}} = \text{fun}(X_i)$ // the optimal particle

6) while $t \leqslant T$ and $\text{fun}(x) > \text{AC}$ do

7) calculate the fitness $\text{fit}_i(t)$

8) updated $G'(t)$, $M_i(t)$, $\text{best}(t)$, $\text{worst}(t)$ according to Eq.(18), Eq.(10) ~ Eq.(12)

9) calculate the $F^d_i(t)$, $a^d_i(t)$ and $v^d_i(t)$

10) update the position of particle according to Eq.(16)

11) calculate the S

12) if $0.9 \leqslant S \leqslant 1$ then

13) use the Tent chaos search

14) end if

15) $t = t + 1$

16) $g_{\rm best} = X_i$ and $f_{\rm best} = {\rm fun}(X_i)$

17) end while

18) return $g_{\rm best}$

5 仿真结果与分析

5.1 实验设计与测试函数

为验证本文算法的可行性和优越性,对10个不同类型的基准测试函数进行仿真实验. 如表1所示,选择4个高维单峰函数$F_1 \sim F_4$,4个高维多峰函数$F_5 \sim F_8$,同时选择2个低维函数F_9和F_{10}. 其中高维函数的最优值为0,因为此类测试函数更能有效地验证ITC-GSA算法的寻优能力[17-19].

表1 基准测试函数

类型	基准测试函数	搜索空间	最优解
高维单峰	$F_1(X) = \sum_{i=1}^{n} x_i^2$	$[-100, 100]^n$	0
	$F_2(X) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	$[-10, 10]^n$	0
	$F_3(X) = \sum_{i=1}^{n} \Big(\sum_{j=1}^{i} x_i \Big)^2$	$[-100, 100]^n$	0
	$F_4(X) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	$[-30, 30]^n$	0
高维多峰	$F_5(X) = \sum_{i=1}^{n-1} [x_i^2 - 10\cos(2\pi x_i) + 10]$	$[-5.12, 5.12]^n$	0
	$F_6(X) = -20\exp\Big(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2} - \exp\Big(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\Big)\Big) + 20 + {\rm e}$	$[-32, 32]^n$	0
	$F_7(X) = \frac{1}{4\,000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\Big(\frac{x_i}{\sqrt{i}}\Big) + 1$	$[-600, 600]^n$	0
	$F_8(X) = 0.1\Big\{ \sin^2(3\pi x_i) + \sum_{i=1}^{n} (x_i - 1)^2 [1 + \sin^2(3\pi x_i + 1)] + (x_n - 1)^2 [1 + \sin^2(2\pi x_n)] \Big\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	$[-50, 50]^n$	0
低维函数	$F_9(X) = \Big(\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6}\Big)^{-1}$	$[-65.53, 65.53]^2$	1
	$F_{10}(X) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4_2^2 + 4x_2^4$	$[-5, 5]^4$	$-1.031\,6$

5.2 算法性能对比分析

在Intel®Core$^{\rm TM}$ i5-3337U CPU@1.80 GHz,内存8.00 G, Windows 7系统和Matlab 2015b下对所提算法进行仿真实验,并与GSA和BPSOGSA算法[20]进行对比.

实验中取种群规模$N = 50$,最大迭代次数$T = 1\,000$,引力常数$G_0 = 100$,衰减速率$\alpha = 20$,对于高维函数维度$D = 30$,对于低维函数F_9和F_{10}的维度D分别为2和4. 为避免偶然性产生的对比误差,3种对比算法对10个基准测试函数分别独立运行50次. 比较各算法的平均值(mean)和标准差(std),结果如表2所示.

由表2可知,对于高维单峰函数F_1和F_2,ITC-GSA无论是在寻优速率还是寻优精度上都比GSA和BPSOGSA有极大的提升,且多次寻优的平均值和标准差也都高出多个数量级,ITC-GSA对函数F_1寻优结果的平均值(mean)相较于GSA提升了17个数量级,相较于BPSOGSA提升12个数量级,同时收敛速度也都比GSA和BPSOGSA快速. 但是对于F_3和F_4,无论是BPSOGSA还是ITC-GSA都未能有所提升. 对于高维多峰函数F_6和F_8,ITC-GSA相较于GSA提升了6个数量级,相较于BPSOGSA提升了4个数量级. 同样对于F_5和F_7,性能提升都不明显. 对于低维函数F_9和F_{10},ITC-GSA相较于GSA和BPSOGSA在寻优精度上都有一定程度的提升.

表 2　算法寻优结果对比

类型	函数	指标	GSA	BPSOGSA	ITC-GSA
高维单峰	F_1	mean	1.63e-17	2.46e-22	1.23e-34
		std	1.16e-18	1.27e-23	1.01e-34
	F_2	mean	2.62e-08	5.23e-10	3.55e-19
		std	1.26e-08	4.24e-11	1.16e-21
	F_3	mean	2.30e+02	5.51e+02	3.08e+02
		std	5.13e+02	4.05e+02	2.13e+01
	F_4	mean	2.61e+01	2.46e+01	2.77e+01
		std	2.86e+01	5.35e+01	5.26e+01
高维多峰	F_5	mean	1.09e+01	8.58e+02	9.95e+00
		std	3.32e+01	6.83e+02	1.15e+01
	F_6	mean	3.82e-09	2.62e-11	4.00e-17
		std	1.27e-10	1.57e-25	2.58e-20
	F_7	mean	3.34e+00	6.26e+00	1.72e+00
		std	2.65e+01	7.33e+01	1.07e+01
	F_8	mean	1.89e-18	3.79e-20	4.05e-33
		std	1.02e-19	6.26e-21	1.07e-35
低维函数	F_9	mean	3.06e+00	2.03e+00	9.98e-01
		std	5.15e+01	4.27e+01	1.27e+00
	F_{10}	mean	−1.03e+00	−1.03e+00	−1.03e+00
		std	4.56e-16	1.49e-07	2.38e-16

由于篇幅有限,但为直观显示GSA、BPSOGSA、ITC-GSA的寻优过程,本文选取这3个算法对函数F_1、F_2、F_6、F_8、F_9的寻优过程曲线,如图7～图11所示. 对于函数F_1、F_2、F_6,ITC-GSA无论是在收敛速度还是在寻优精度上都明显优于GSA和BPSOGSA,且迭代前期的搜索性能和迭代末期的开拓性能也都优于GSA和BPSOGSA,在保证开拓能力的同时也充分保证搜索能力,不失多样性和稳定性. 虽然在末期有陷于局部最优的趋势,但是由于引入了Tent混沌搜索,粒子种群能够有效地跳出局部最优,从而提高了寻优精度. 对于函数F_2和F_6,跳出局部最优的现象比较明显,在迭代末期都曾多次跳出局部最优,体现出ITC-GSA具有良好的搜索能力. 对于函数F_8,ITC-GSA的收敛速度和寻优精度同样优于GSA和BPSOGSA. 但是,对于低维函数F_9,则未能产生明显的优化效果.

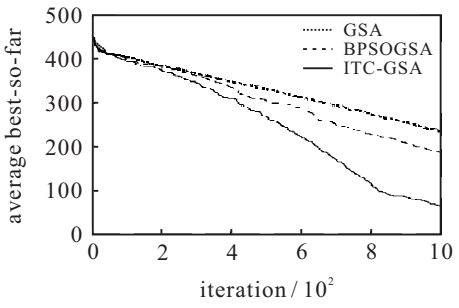

图 7　GSA、BPSOGSA、ITC-GSA对F_1的寻优过程曲线

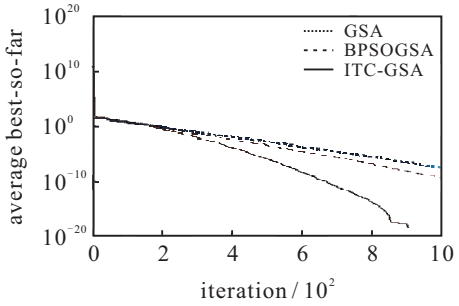

图 8　GSA、BPSOGSA、ITC-GSA对F_2的寻优过程曲线

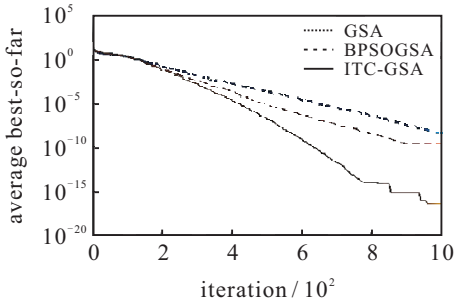

图 9　GSA、BPSOGSA、ITC-GSA对F_6的寻优过程曲线

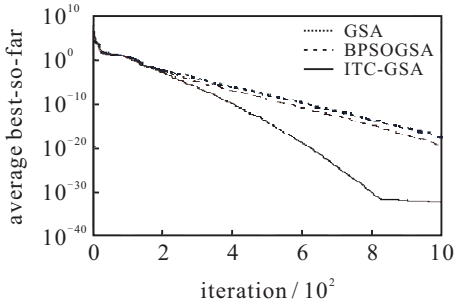

图 10　GSA、BPSOGSA、ITC-GSA对F_8的寻优过程曲线

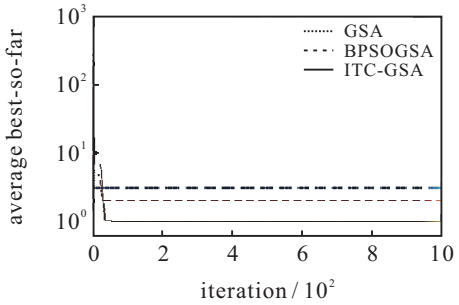

图 11　GSA、BPSOGSA、ITC-GSA对F_9的寻优过程曲线

综上所述,ITC-GSA虽然对函数F_3、F_4、F_5、F_7的寻优性能较GSA和BPSOGSA不明显,但对于函数F_1、F_2、F_6、F_8、F_9,寻优性能都高出多个数量级,收敛速度也明显好于GSA和BPSOGSA,这充分说明了ITC-GSA的可行性和优越性.

6　结　论

本文所提的ITC-GSA算法在种群初始化时引入混沌序列,采用Tent混沌保证了初始化种群的遍历性、均匀性和随机性. 加入自适应引力常数和Tent混沌搜索,有效控制了算法的收敛速度,提高了寻优精度,且当粒子陷入局部最优时能够有效跳出局部

最优,提高寻优精度. 最后,通过理论和对比实验验证了本文算法的有效性和优越性. 该算法较GSA和BPSOGSA具有很大的优势,在收敛速度和寻优精度上都有明显的提高,体现出优良的搜索性能和开拓性能. ITC-GSA的理论和研究还在初始阶段,后期将把该算法应用于实践工程中,如TSP问题、NP问题、软件测试用例生成[21]、测试用例优先级调整[22]等领域,扩展ITC-GSA算法的应用场景.

参考文献(References)

[1] Rashedi E, Nezamabadi-Pour H, Saryazdi S. GSA: A gravitational search algorithm[J]. Information Sciences, 2009, 179(13): 2232-2248.

[2] Beigvand S D, Abdi H, Scala M L. Combined heat and power economic dispatch problem using gravitational search algorithm[J]. Electric Power Systems Research, 2016, 133: 160-172.

[3] Chen Z, Yuan X, Tian H, et al. Improved gravitational search algorithm for parameter identification of water turbine regulation system[J]. Energy Conversion & Management, 2014, 78(30): 306-315.

[4] Shen D, Jiang T, Chen W, et al. Improved chaotic gravitational search algorithms for global optimization[C]. IEEE Congress on Evolutionary Computation(CEC). Sendai: IEEE, 2015: 1220-1226.

[5] Li C, Zhou J, Xiao J, et al. Parameters identification of chaotic system by chaotic gravitational search algorithm[J]. Chaos Solitons & Fractals, 2012, 45(4): 539-547.

[6] Han X H, Xiong X, Fu D. A new method for image segmentation based on BP neural network and gravitational search algorithm enhanced by cat chaotic mapping[J]. Applied Intelligence, 2015, 43(4): 855-873.

[7] Gao S, Vairappan C, Wang Y, et al. Gravitational search algorithm combined with chaos for unconstrained numerical optimization[J]. Applied Mathematics & Computation, 2014, 231(11): 48-62.

[8] Liu L, Sun S Z, Yu H, et al. A modified Fuzzy C-Means (FCM) Clustering algorithm and its application on carbonate fluid identification[J]. Journal of Applied Geophysics, 2016, 129: 28-35.

[9] 匡芳君, 徐蔚鸿, 金忠. 自适应Tent混沌搜索的人工蜂群算法[J]. 控制理论与应用, 2014, 31(11): 1502-1509.
(Kuang F J, Xu W H, Jin Z. Artificial bee colony algorithm based on self-adaptive Tent chaos serch[J]. Control Theory & Applications, 2014, 31(11): 1502-1509.)

[10] 单梁, 强浩, 李军,等. 基于Tent映射的混沌优化算法[J]. 控制与决策, 2005, 20(2): 179-182.
(Shan L, Qiang H, Li J, et al. Chaotic optimization algorithm based on Tent map[J]. Control and Decision, 2005, 20(2): 179-182.)

[11] Zibanezhad B, Zamanifar K, Sadjady R S, et al. Applying gravitational search algorithm in the QoS-based Web service selection problem[J]. Journal of Zhejiang University: Science, 2011, 12(9): 730-742.

[12] Khajehzadeh M, Taha M R, El-Shafie A, et al. A modified gravitational search algorithm for slope stability analysis[J]. Engineering Applications of Artificial Intelligence, 2012, 25(8): 1589-1597.

[13] Moghadam M S, Nezamabadi-Pour H, Farsangi M M. A quantum behaved gravitational search algorithm[J]. Intelligent Information Management, 2012, 4(6): 390-395.

[14] Sahu R K, Panda S, Padhan S. A novel hybrid gravitational search and pattern search algorithm for load frequency control of nonlinear power system[J]. Applied Soft Computing, 2015, 29: 310-327.

[15] Jiang S, Ji Z, Shen Y. A novel hybrid particle swarm optimization and gravitational search algorithm for solving economic emission load dispatch problems with various practical constraints[J]. International Journal of Electrical Power & Energy Systems, 2014, 55(2): 628-644.

[16] Mirjalili S, Lewis A. Adaptive gbest-guided gravitational search algorithm[J]. Neural Computing & Applications, 2014, 25(7/8): 1569-1584.

[17] 肖辉辉, 万常选, 段艳明, 等. 基于引力搜索机制的花朵授粉算法[J]. 自动化学报, 2017, 43(4): 576-594.
(Xiao H H, Wan C X, Duan Y M, et al. Flower pollination algorithm based on gravity search mechanism[J]. Acta Automatica Sinica, 2017, 43(4): 576-594.)

[18] Mirjalili S, Wang G G, Coelho L D S. Binary optimization using hybrid particle swarm optimization and gravitational search algorithm[J]. Neural Computing & Applications, 2014, 25(6): 1423-1435.

[19] Güvenç U, Katrcolu F. Escape velocity: A new operator for gravitational search algorithm[J]. Neural Computing & Applications, 2017(13): 1-16.

[20] Biswas S, Kundu S, Das S. Inducing niching behavior in differential evolution through local information sharing[J]. IEEE Transactions on Evolutionary Computation, 2015, 19(2): 246-263.

[21] Bao X A, Yang Y J, Zhang N, et al. Test case generation method based on adaptive particle swarm optimization[J]. Computer Science, 2017, 44(6): 177-181.

[22] Zhang N, Yao L, Bao X A, et al. Multi-objective optimization based on-line adjustment strategy of test case prioritization[J]. Journal of Software, 2015, 26(10): 2451-2464.

作者简介

张娜(1977−), 女, 副教授, 硕士, 从事软件工程、软件测试等研究, E-mail: zhangna@zstu.edu.cn;

赵泽丹(1991−), 男, 硕士, 从事软件测试、智能算法、智能信息处理的研究, E-mail: 485808850@qq.com;

包晓安(1973−), 男, 教授, 博士, 从事云计算、自适应软件、智能信息处理等研究, E-mail: baoxiaoan@zstu.edu.cn;

钱俊彦(1973−), 男, 教授, 博士, 从事软件工程、软件分析、云计算等研究, E-mail: qjy2000@gmail.com;

吴彪(1989−), 男, 博士, 从事计算机及计算机应用、自动化技术的研究, E-mail: w501sn@yamaguchi-u.oc.jp.

(责任编辑: 齐 霁)