改进的细菌觅食优化算法求解 0-1 背包问题

杜明煜,雷秀娟
(陕西师范大学 计算机科学学院,陕西 西安 710062)

摘 要:细菌觅食优化算法作为一种新兴的智能优化算法,一般用来解决连续域的问题。为了解决离散域问题,提出了一种改进的细菌觅食优化算法。采用线性递减的思想和随机的游动长度代替固定步长和随机游动方向,改进了趋向性操作方案,并将其应用于解决 0-1 背包问题。将改进的细菌觅食优化算法与遗传算法、离散粒子群优化算法及基本的离散化细菌觅食优化算法分别在小规模和大规模的 0-1 背包问题上进行了仿真比较,表明了改进的细菌觅食优化算法能取得较好的效果,寻优能力强。

关键词:0-1 背包;离散域;细菌觅食优化算法
中图分类号:TP301 **文献标识码**:A **文章编号**:1673-629X(2014)05-0044-04
doi:10.3969/j.issn.1673-629X.2014.05.011

Improved Bacteria Foraging Optimization Algorithm for 0-1 Knapsack Problem

DU Ming-yu, LEI Xiu-juan
(School of Computer Science, Shaanxi Normal University, Xi'an 710062, China)

Abstract: As a new intelligent optimization algorithm, Bacteria Foraging Optimization (BFO) algorithm is generally used to solve continuous problems. In order to solve discrete problems, an improved BFO algorithm was proposed in this paper, which adopted linearly decreasing thoughts and random steps of the bacterial tumble instead of fixed steps and random direction to promote chemotaxis solution, and applied to 0-1 knapsack problem. By contrast with Genetic Algorithms (GA), Discrete Particle Swarm Optimization (DPSO) and basic BFO algorithm on both small-scale and large-scale 0-1 knapsack problems, the simulation results indicate that the improved BFO algorithm performs better with high search capability.

Key words: 0-1 knapsack; discrete domain; bacteria foraging optimization

0 引 言

0-1 背包问题是组合优化问题中经典的 NP 问题,在资源分配、信息加密、预算控制、项目选择、材料切割、货物装载等问题中具有重要价值,是一个经典的 NP 难问题,该问题一直是算法和复杂性研究的热点问题之一[1]。

细菌觅食优化算法(Bacteria Foraging Optimization, BFO)是 2002 年 Passino 提出的一种仿生随机搜索算法,该算法模拟人类大肠杆菌觅食行为,尽量避免不利于寻找食物的自然环境,向食物浓度高的环境移动,在觅食过程中,淘汰觅食能力差的个体,觅食能力强的细菌分裂出相同的两个细菌,提高了寻找最优解的速度。具有群体智能算法的并行搜索、易跳出局部极小值等优点。

该算法主要通过趋向性操作、复制操作和迁徙操作中迭代计算来求解[2-6]。

BFO 主要应用于资源分配和调度[7-9]、预测控制[10-11]、图形图像[12-13]、电力系统控制[14-16]等领域,但对于离散域的问题,主要使用的是遗传算法[17]、离散粒子群优化算法[18-19]等,文中提出了一种改进的 BFO 算法来求解离散域中 0-1 背包问题。

收稿日期:2013-07-23 **修回日期**:2013-10-26 **网络出版时间**:2014-02-11
基金项目:国家自然科学基金青年基金(61100164,61173190);教育部留学回国人员科研启动基金(教外司留[2012]1707 号);中央高校基本科研业务费专项资金项目(GK201302025);陕西省 2010 年自然科学基础研究计划青年基金(2010JQ8034)
作者简介:杜明煜(1989-),女,山东临沂人,硕士研究生,研究方向为智能优化算法;雷秀娟,博士,教授,硕士生导师,CCF 会员,ACM 会员,通讯作者,研究方向为智能计算与智能优化、生物信息计算等。
网络出版地址:http://www.cnki.net/kcms/detail/61.1450.TP.20140211.1717.061.html

1　细菌觅食优化算法的基本原理

1.1　趋向性操作

大肠杆菌在觅食过程中有两种不同形式的运动:游动和旋转。通常,细菌在食物浓度高的地区会较多地运动;在食物浓度较低的环境会频繁地旋转。一个细菌所处的位置表示一个问题的可行解,假设细菌种群规模为 S, $\theta^i(j,k,l)$ 表示细菌 i 在第 j 次趋向性操作、第 k 次复制操作和第 l 次迁徙操作之后的位置。细菌 i 的每一步趋向性操作可以表示为

$$\theta^i(j+1,k,l)=\theta^i(j,k,l)+C(i)\varphi(j) \tag{1}$$

其中, $C(i)>0$ 表示向前移动的步长单位; $\varphi(j)$ 表示旋转后选择的随机前进方向。

细菌的前进可以看作是解空间中对可行解的搜索,减速意味着对当前搜索得到的解不够满意,准备对其调整,停下来原地摆动对应着对当前解的优劣的判断,判断可以通过适应度函数进行,判断结束后决定是否对当前解进行调整及如何调整。

1.2　复制操作

为了保持整个群体规模的恒定,一部分觅食能力强的细菌进行分裂,分裂出的细菌与母体完全相同,以替换觅食能力差的细菌。复制操作是根据已知的适应度值对群体中的每个个体所代表的可行解进行优劣性评估,保留优秀个体,并对优秀个体进行复制以代替劣质个体。通过复制操作,群体中优良个体得到保护,竞争能力弱的不良个体被淘汰,同时也能产生更优的个体,既提高了寻找最优解的速度,又保持了群体的多样性。设淘汰细菌个数为 $S/2$,剩下的 $S/2$ 进行自我复制,即生成的新个体与原个体具有相同的位置。

1.3　迁徙操作

细菌个体生活的区域可能会突然发生变化导致细菌种群集体死亡或者集体迁徙到一个新的生活区域,细菌觅食优化算法的迁徙操作是按照给定概率发生的,如果某个细菌满足迁徙发生的概率,这个细菌灭亡,并且随机地在问题解空间任意位置产生一个新的个体。迁徙操作使得细菌具有随机搜索能力,有助于保持种群的稳定性和多样性,跳出局部最优解,减少早熟收敛的情况。

2　改进的细菌觅食优化算法在 0-1 背包问题上的应用

研究发现,标准的细菌觅食优化算法采用的是固定步长,不利于算法的收敛,不能很好地找到最优解,收敛速度不灵活[20]。现阶段细菌觅食优化算法一般用来解决连续域的问题,而若使用 sigmoid 函数进行离散编码[19],虽然可以将问题编码离散化,但是时间耗费大。文中提出了一种改进的编码方法,采用了线性递减的思想和可变的随机游动长度,显著提高了算法的运行效率和求解质量,并在求解 0-1 背包问题上取得了很好的效果。

2.1　问题描述

给定 n 种物品和一个背包,物品 i 的重量是 w_i,其价值为 P_i,背包的容量为 V。问应如何选择装入背包中的物品使得背包中物品的价值量最大,选择物品时只能选择装入或者不装入。数学模型可以表示为:

$$\max \sum_{i=1}^{n} P_i x_i$$

$$\text{s.t.} \begin{cases} \sum_{i=1}^{n} w_i x_i \leqslant V \\ x_i \in \{0,1\}, 1 \leqslant i \leqslant n \end{cases}$$

2.2　编码方法

每个细菌的位置 $\boldsymbol{\Theta}$ 用维度与物品个数 n 相同的二进制编码表示, $\boldsymbol{\Theta}=(\theta_1,\theta_2,\cdots,\theta_n)$。其中, θ_i 为 0 代表第 i 个物品不放入背包, θ_i 为 1 代表第 i 个物品放入背包。每个细菌的位置代表一种物品放入方式,是一个可行解,细菌的维度为物品数目,细菌觅食过程是寻找最优解的过程。

2.3　适应度函数

对于 0-1 背包问题而言,希望找到一个组合使得放入背包中物品的价值量最大。因此,适应度函数定义为放入背包中物品的价值。在细菌觅食算法中表现为细菌的位置 $\boldsymbol{X}$ 与物品价值向量 $\boldsymbol{P}$ 的乘积。适应度函数值越大,放入背包中物品的价值量越高。

2.4　趋向性操作

在解决组合优化问题时,一般希望算法前期有较高的全局搜索能力,以扩展搜索空间;算法后期有较高的局部搜索能力,以加快收敛速度。分析结果表明[5],步长大,全局搜索能力强;反之,局部搜索能力强。因此算法前期应该设置大步长,而后期应该设置小步长。

对于细菌的趋向性操作,采用权重线性递减的思想[7],生成一个长度线性递减的数字作为步长 C,步长调整公式为:

$$C=\left\lceil \frac{\text{iter}_{\max}-\text{iter}}{\text{iter}_{\max}} \times n \right\rceil \tag{2}$$

其中, $\text{iter}_{\max}$、iter 分别是迭代次数最大值和当前迭代次数; n 是物品数量。

随机生成长度为 C 的二进制字符串 Cb,当随机数大于某个设定的值 Pc 时,将二进制的位置编码前 C 项中值为 0 的字符用步长串中的字符替换,模拟细菌的向前运动;否则,将二进制位置编码前 C 项中值为 1 的字符用步长串中的字符替换,模拟向后运动。细菌 i 的每一步趋向性操作可以表示为

$$\theta^i(j+1,k,l)=\begin{cases}\theta^{\prime i}(j,k,l)+\mathrm{Cb}\ \ \mathrm{rand}>\mathrm{Pc}\\ \theta^{\prime\prime i}(j,k,l)-\mathrm{Cb}\ \ \mathrm{else}\end{cases}\quad(3)$$

其中，$\theta^{\prime i}(j,k,l)$ 和 $\theta^{\prime\prime i}(j,k,l)$ 分别表示 $\theta^i(j,k,l)$ 前 C 项中值为 0、1 的字符串。

随着迭代次数 iter 的增加，步长 C 变小。采用线性递减的思想和可变的随机游动长度的离散化方法，使得细菌趋向性操作同时进行了方向和游动长度的改变，提高了算法的运行效率，使得算法前期具有较强的全局搜索能力，后期具有较强的局部搜索能力。

2.5 复制操作

将细菌位置按照适应度函数值排序，对于每一个处于后 50% 的细菌位置，随机地用前 50% 中的一个细菌位置来替换，使得优秀的个体得到保存，淘汰觅食能力差的细菌，同时又不失细菌多样性。

2.6 算法流程

改进的细菌觅食优化算法求解 0-1 背包问题步骤：

Step1：初始化细菌种群规模 S，细菌初始位置，趋向性操作、复制操作、迁徙操作的执行次数，趋向性操作的最大游动长度、方向概率 Pc，$j=0,k=0,l=0$。

Step2：若超过迁徙操作的最大执行次数，则结束。

Step3：若超过复制操作的最大执行次数，则进行迁徙操作，$l=l+1$，转向 Step2。

Step4：若超过趋向性操作的最大执行次数，则进行复制操作，$k=k+1$，转向 Step3。

Step5：使用公式(2)、公式(3)执行趋向性操作，$j=j+1$，转向 Step4。

2.7 算法复杂度分析

BFO 算法由趋向性操作、复制操作、迁徙操作三重循环构成，趋向性操作分为对每一维方向的改变和对每一维移动步长的改变；改进的 BFO 算法在趋向性操作上对变量整体进行改变，同时改变了方向和步长。

假设算法中变量的维数为 d，细菌数目为 S，趋向性操作、复制操作、迁徙操作的执行次数分别为 nc，nre，nt，细菌的趋向性操作前进的最大步长为 ns，最大迭代次数为 $\mathrm{iter}_{\max}$，则 BFO 算法的算法复杂度为 $\Theta(\mathrm{iter}_{\max}\times d\times s\times\mathrm{nc}\times\mathrm{nre}\times\mathrm{nt}\times\mathrm{ns})$，而改进的 BFO 算法的算法复杂度为 $\Theta(\mathrm{iter}_{\max}\times s\times\mathrm{nc}\times\mathrm{nre}\times\mathrm{ns})$。算法的最大迭代次数、细菌数目、维数都是给定的，因此算法主要取决于 nc，nre，nt，ns。

3 仿真实验

算法通过 Matlab7.13 编程实现，在内存为 4 G，CPU 速度为 3.10 GHz 的双核计算机上运行。分别采用精英保留策略的 GA[17]、DPSO[18]、采用 sigmoid 函数[18]的离散化 DBFO 算法、改进的 BFO 优化算法做 30 次独立的实验进行仿真比较，求得多次实验结果的最优解、平均解、平均运行时间。

3.1 参数设置

各算法的相关参数设置如下：所有算法的最大迭代次数 $\mathrm{iter}_{\max}$ 均为 200。对于基于精英保留策略的遗传算法，种群规模为 40，交叉概率为 0.8，变异概率为 0.085，精英保留个数为 3；对于离散粒子群优化算法，种群规模为 40，$c_1=0.4$，$c_2=0.9$；对于细菌觅食优化算法，种群规模为 20，趋向性操作、复制操作、迁徙操作的执行次数分别为 10、4、2，细菌最大游动长度为 4，迁徙概率为 0.025。分别采用背包规模为 50、200 的背包 1、背包 2，相关参数设置如表 1、表 2。

表 1 背包规模为 50 的背包 1 参数表

物品重量 w_i	物品价值 P_i	背包容量 V
2 5 1 9 3 6 4 9 3 2 8 9 6 2 5 2 5 9 4 2 3 4 8 5 4 3 1 2 1 1 3 3 6 3 1 2 1 4 1 1 4 5 3 5 5 2 6 1 5 3	3 13 10 13 5 6 6 3 11 3 9 2 7 9 7 4 6 3 9 4 9 5 8 9 10 14 7 8 7 9 5 5 11 7 5 11 6 9 8 9 11 8 5 10 10 9 10 8 10 12	80

表 2 背包规模为 200 的背包 3 参数表

物品重量 w_i	物品价值 P_i	背包容量 V
8 20 18 8 8 10 20 13 15 18 8 12 18 3 18 3 18 7 12 11 1 5 2 4 6 4 18 9 7 18 19 15 1 3 11 20 17 20 4 6 7 3 13 17 17 4 2 1 1 4 7 20 10 1 19 20 5 11 12 1 7 3 10 6 20 11 13 2 20 1 4 18 18 20 6 12 12 1 12 19 15 16 5 8 6 2 2 1 6 6 15 8 11 12 14 3 16 6 15 19 20 9 4 16 3 14 5 3 17 19 15 10 2 9 7 13 13 3 9 1 6 20 8 15 8 17 19 6 20 17 1 20 11 12 4 10 15 2 7 17 10 6 12 4 4 12 2 20 6 5 13 12 5 5 19 4 19 17 2 17 12 16 18 2 18 14 12 1 10 6 10 2 18 20 1 8 7 1 14 7 5 17 5 13 9 3 11 19 10 7 9 1 15 17 11 15 19 7 4 4	98 42 27 4 41 85 38 52 26 12 44 87 92 45 95 23 44 48 75 20 57 25 66 62 90 31 9 97 81 83 54 74 92 54 88 81 70 96 44 84 36 74 50 38 27 58 82 50 40 2 25 71 65 21 14 50 59 34 60 38 42 17 69 80 76 38 91 58 85 38 75 91 27 39 80 90 72 32 59 80 49 66 54 20 88 33 68 21 98 58 86 38 43 61 13 88 27 41 44 68 18 59 32 17 5 86 23 47 95 46 22 35 54 11 9 40 61 41 11 8 34 2 4 100 28 54 16 58 44 45 67 23 10 44 84 22 61 13 54 14 52 81 91 40 63 73 76 67 89 19 61 40 3 15 51 69 89 36 42 46 9 55 57 69 98 63 41 46 2 5 89 16 64 49 77 53 76 70 95 87 82 26 19 33 92 83 78 83 92 3 63 59 89 82 45 5 46 1 33 54	1 000

3.2 实验结果与分析

由图 1 可以看出，对于背包规模为 50 的背包 1 问题的四种优化算法相比，由于问题规模较小，GA 的收

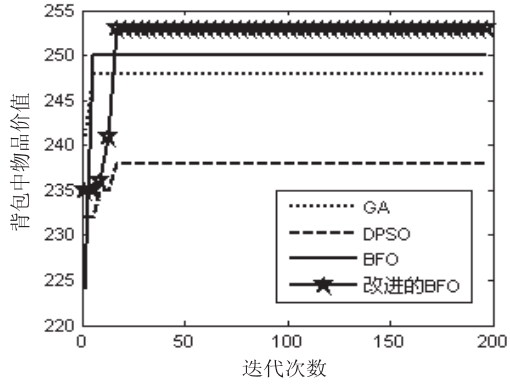

图 1 四种算法求得背包 1 问题的最优解

敛速度最快,在极短的时间内就可以找到极值;DPSO 收敛速度最慢,得到的物品价值明显比其他三种算法要低;改进的 BFO 寻优能力最强。由图 2 可知,当背包规模增加到200 时,DPSO 求得的价值量最低,GA 的收敛速度最快,改进的 BFO 得到的价值量最大。

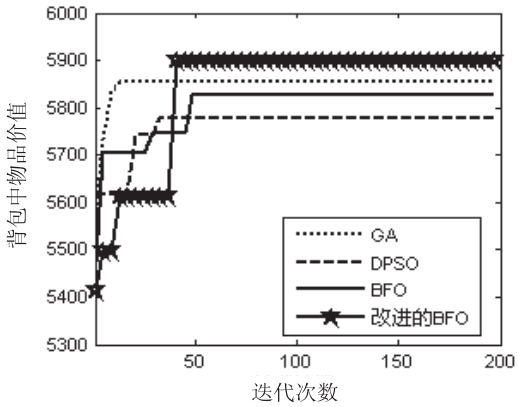

图 2　四种算法求得背包 2 问题的最优解

由表 3、表 4 可知,无论是小规模还是大规模的问题,从运行时间上看,GA 的运行速度最快,改进的 BFO 运行时间比 DPSO 稍慢,BFO 的运行时间最长;从求得的最优解和平均解可知,改进的 BFO 无论是最优解还是平均解都是四种算法中最好的。

表 3　四种算法求解背包 1 问题的性能比较

算法	最优解	平均解	运行时间/s
GA	248	247.940 0	0.156 0
DPSO	238	237.655 0	1.560 0
BFO	250	249.775 0	71.713 7
改进的 BFO	253	251.880 0	3.853 2

表 4　四种算法求解背包 2 问题的性能比较

算法	最优解	平均解	运行时间/s
GA	5 859	5 817.167 0	0.811 2
DPSO	5 774	5 687.666 7	5.257 2
BFO	5 831	5 756.450 0	178.327 7
改进的 BFO	5 902	5 863.850 0	7.020 0

综合图表来看,GA 的收敛速度最快,取得的效果适中;DPSO 最后粒子虽然都趋于所找到的最优解,但得到的最优解明显没有 BFO 和改进的 BFO 算法好。GA 算法由于具有隐并行性,因此其运行时间最短;BFO 算法由于 sigmoid 函数将每一维变量离散化,趋向性操作与游动计数器有关,而且经过趋向性操作、复制操作、迁徙操作三重循环,因此花费的时间代价是最大的;改进的 BFO 采用了线性递减的思想和可变的随机游动长度,省略了对每一维的单独处理和游动计数器循环,使得对每一个可行解整体操作,大大缩短了运行时间,因此,改进的 BFO 算法的运行时间比基于 sigmoid 函数的离散化方法运行时间低,但是由于趋向性操作、复制操作、迁徙操作的循环次数比较大,因此比 DPSO 要略微长一点,但是寻优能力最强。

4　结束语

文中提出了一种改进的细菌觅食优化算法来解决 0-1 背包问题,采用线性递减的思想搭配随机的游动长度改进 BFO 趋向性操作的离散化,使得细菌趋向性操作同时进行了方向和游动长度的改变,与标准的细菌觅食优化算法相比,减少了一层循环操作,因此缩短了运行时间,提高了算法的运行效率;游动长度和方向随着迭代次数增加而变小,使得算法具备较好的全局搜索能力和局部探索能力。

参考文献:

[1] Asho I. Interactive knapsacks: theory and applications[D]. Tampere: University of Tampere, 2002.

[2] 王晓东. 计算机算法设计与分析[M]. 北京:电子工业出版社,2007.

[3] Passino K M. Biomimicry of bacterial foraging for distributed optimization and control[J]. IEEE Control Systems, 2002, 22(3): 52-67.

[4] Passino K M. Biomimicry for optimization, control, and automation[M]. Berlin: Springer, 2005.

[5] 雷秀娟. 群智能优化算法及其应用[M]. 北京:科学出版社,2012:70-74.

[6] 周雅兰. 细菌觅食优化算法的研究与应用[J]. 计算机工程与应用,2010,46(20):16-21.

[7] Muñoz M A, López J A, Caicedo E F. Bacteria swarm foraging optimization for dynamical resource allocation in a multizone temperature experimentation platform[J]. Advances in Soft Computing, 2007, 41: 427-435.

[8] 崔静静,孙延明,车兰秀. 改进细菌觅食算法求解车间作业调度问题[J]. 计算机应用研究,2011,28(9):3324-3326.

[9] 吴　波,郝春梅. 细菌觅食优化算法在嵌入式系统多任务调度中的应用[J]. 微电子学与计算机,2013,30(3):143-147.

[10] Majhi R, Panda G, Majhi B, et al. Efficient prediction of stock market indices using adaptive bacterial foraging optimization (ABFO) and BFO based techniques[J]. Expert Systems with Applications, 2009, 36(6): 10097-10104.

[11] 王雪松,程玉虎,郝名林. 基于细菌觅食行为的分布估计算法在预测控制中的应用[J]. 电子学报,2010,38(2):333-339.

[12] Panda R, Naik M K, Panigrahi B K. Face recognition using bacterial foraging strategy[J]. Swarm and Evolutionary Computation, 2011, 1(3): 138-146.

[13] Verma O P, Hanmandlu M, Sultania A K, et al. A novel fuzzy system for edge detection in noisy image using bacterial forag-

(下转第 52 页)

表 1 计算结果的精度评定

统计项	最小差值	最大差值	平均值	均方差	标准差
南北分量(″)	-4.07	3.17	-0.10	1.35	1.14
东西分量(″)	-5.71	3.98	-0.06	1.84	1.34

计算结果充分说明了此方法的可行性。

4 结束语

文中通过介绍一种 Fortran 和 C#混合编程的方法,来说明混合编程在卫星测高中的应用。通过编写的计算垂线偏差的软件计算了两个 cycle 的南海海域的垂线偏差值,计算结果达到了理想的精度。充分说明了混合编程在卫星测高中的优势:

(1)可以大批量、自动化地处理繁多的卫星测高数据,减少使用人员的工作量。

(2)在混合编程的核心计算算法是用 Fortran 编写的,这些算法相互独立调用,便于软件维护和升级。

(3)现有很多的卫星测高的算法都是利用 Fortran 编写的,通过混合编程的方法可将这些算法集成到软件中,便于代码的重用,以及降低软件开发的难度。

参考文献:

[1] 彭国伦. Fortran 95 程序设计[M]. 北京:中国电力出版社,2011.

[2] Watson K,Nagel C. C#入门经典[M]. 第 5 版. 齐立波,译. 北京:清华大学出版社,2012.

[3] 李建成,宁津生,晁定波,等. 卫星测高在大地测量学中的应用及进展[J]. 测绘科学,2006,31(6):19-23.

[4] 周振红,任 慧,杜丽平. Fortran DLL 组件集成到. NET 平台(一)[J]. 武汉大学学报(工学版),2005,38(4):100-103.

[5] 周振红,毕苏萍,张成才. Fortran DLL 组件集成到. NET 平台(二)[J]. 武汉大学学报(工学版),2006,39(6):51-54.

[6] 周振红,吴虹娟,颜国红. Fortran 与 Visual C++混合编程研究[J]. 武汉大学学报(工学版),2001,34(2):84-87.

[7] 周振红,宋宇伟,郭恒亮,等. Visual FORTRAN 基于 Win32 DLL 的混合编程技术[J]. 郑州大学学报(工学版),2003,24(3):10-13.

[8] Hwang C,Parsons B. Gravity anomalies derived from Seasat,Geosat,ERS-1 and TOPEX/POSEIDON altimetry and ship gravity:A case study over the Reykjanes Ridge[J]. Geophysical Journal International,1995,122(2):551-568.

[9] Olgiati A,Balmino G,Sarrailh M,et al. Gravity anomalies from satellite altimetry:comparison between computation via geoid heights and via deflections of the vertical[J]. Bulletin Geodesique,1995,69(4):252-260.

[10] Sandwell D T. A detailed view of the South Pacific geoid from satellite altimetry[J]. Journal of Geophysical Research:Solid Earth,1984,89(B2):1089-1104.

[11] Sandwell D T. Antarctic marine gravity filed from high-density satellitealtimetry[J]. Geophysical Journal International,1992,109(2):437-448.

[12] Sandwell D T,Smith W H F. Marine gravity anomaly from Geosat and ERS-1 satellite altimetry[J]. Journal of Geophysical Research:Solid Earth,1997,102(B5):10039-10054.

[13] Hwang C. Inverse Vening Meinesz formula and deflection-geoid formula:Applications to the predictions of gravity and geoid over the South China Sea[J]. Journal of Geodesy,1998,72:304-312.

[14] Hwang C,Hsu H Y,Jang R J. Global mean sea surface and marine gravity anomaly from multi-satellite altimetry:Applications of deflection-geoid and inverse Vening Meinesz formulae[J]. Journal of Geodesy,2002,76(8):407-418.

(上接第 47 页)

ing[J]. Multidimensional Systems and Signal Processing,2013,24(1):181-198.

[14] Ray P K,Subudhi B. BFO optimized RLS algorithm for power system harmonics estimation[J]. Applied Soft Computing,2012,12(8):1965-1977.

[15] Flah A,Sbita L. A novel IMC controller based on bacterial foraging optimization algorithm applied to a high speed range PMSM drive[J]. Applied Intelligence,2013,38(1):114-129.

[16] 陈 雷,张立毅,郭艳菊,等. 基于细菌觅食优化的盲信号提取算法[J]. 计算机应用研究,2012,29(2):451-454.

[17] 苏 凯. 基于遗传算法的决策空间离散分布约束优化问题研究[D]. 北京:华北电力大学,2012.

[18] Jordehi A R,Jasni J. Particle swarm optimization for discrete optimization problems:a review[M]//Artificial Intelligence Review. [s. l.]:Springer,2012.

[19] Kennedy J,Eberhart R C. A discrete binary version of the particle swarm algorithm[C]//Proc of IEEE international conference on systems,man,and cybernetics. Orlando,FL:IEEE Press,1997:4104-4108.

[20] Chen Hanning,Zhu Yunlong,Hu Kunyuan. Self-adaptation in bacterial foraging optimization algorithm[C]//Proceedings of 2008 3rd international conference on intelligent system and knowledge engineering. Xiamen:[s. n.],2008:1026-1031.

改进的细菌觅食优化算法求解0-1背包问题

万方数据 WANFANG DATA 文献链接

作者： 杜明煜， 雷秀娟， DU Ming-yu， LEI Xiu-juan
作者单位： 陕西师范大学 计算机科学学院,陕西 西安,710062
刊名： 计算机技术与发展 ISTIC
英文刊名： Computer Technology and Development
年，卷(期)： 2014(5)

本文链接：http://d.g.wanfangdata.com.cn/Periodical_wjfz201405011.aspx