# 一种混合的离散细菌菌落优化算法*

宋德逻，孔德福，李　明†
（西南林业大学 机械与交通学院，昆明 650224）

**摘　要**：为了利用细菌算法解决组合优化问题，提出了一种混合的离散细菌菌落优化算法。根据现有细菌优化算法，设计一种新的个体编码方式及进化模式，通过设计种群的自适应调整因子增强个体活力，并融合禁忌搜索算法，克服算法易于陷入过早收敛的不足，并与其他算法在 Taillard 标准调度测试问题集上比较实验，验证了算法的有效性。仿真结果表明，该算法可以搜索到问题的最优组合，能够有效避免算法陷入局部最优，取得了满意的结果。

**关键词**：离散优化算法；细菌菌落；进化模式；自适应调整
**中图分类号**：TP301.6　　**文献标志码**：A　　**文章编号**：1001-3695(2014)02-0358-03
doi:10.3969/j.issn.1001-3695.2014.02.009

# Hybrid discrete bacterial colony optimization algorithm

SONG De-luo, KONG De-fu, LI Ming†
(*College of Machinery & Transportation, Southwest Forestry University, Kunming* 650224, *China*)

**Abstract**: This paper presented a hybrid discrete bacterial colony optimization algorithm to solve the combination optimization problems, and designed novel coding method and evolution mechanism for each individual. According to such coding way, every individual was assigned a discrete value that represented a feasible combination of the problems. So the whole searching behaviors of the colony were executed in the discrete space directly. In order to improve the performance of the individual, each bacterium was given an adaptive adjustment factor that could adaptively reinitialize the position of individuals when all of them gathered together. This algorithm added the taboo searching to enhance the local searching simultaneously. It verified the performance of the algorithm by some Taillard's benchmark problems. Simulation results show that all optimal solutions can be found in experiments and it can effectively avoid the search being trapped into local optimum and achieve satisfactory result.
**Key words**: discrete optimization algorithm; bacterial colony; evolution pattern; adaptive adjustment

## 0　引言

智能群集优化算法已广泛用于解决各种复杂优化问题，由于相应的基础理论还不够完善，各种新的群集优化算法层出不穷。2002 年美国 Passino 根据大肠杆菌的觅食行为提出了一种细菌觅食优化算法(bacterial foraging optimization, BFO)[1]，并且被应用于解决 PID 控制器参数设计[2]、神经网络权值训练[3]、金融预测[4]等连续函数优化问题。虽然 BFO 算法相对传统的微粒群算法增加了一定的进化机制，但是该算法只是简单地将单个细菌的觅食行为扩展为细菌群体的觅食行为，其中的某些思想和观点与实际细菌菌落的演化过程不符。李明等人[5]通过模拟细菌菌落的演化过程，提出了一种细菌菌落优化算法(bacterial colony optimization algorithm, BCO)，不仅建立了群体生长繁殖的进化机制，还提出了一种算法自然结束的准则。

现有细菌算法主要针对连续函数优化问题，显然无法直接用于解决组合优化问题，为此，文献[6]通过对 LOV 规则(largest-order-value rule)的修改，提出了一种基于工序的编码方案，同时引入自适应步长和差分进化算子，实现对车间作业调度问题寻优，但算法容易出现早熟收敛。文献[7]对 BFO 算法的趋化性进行改进，提出了基于细菌个体和群体信息的两种搜索策略求解车间作业问题，但算法使用单一的领域搜索，无法对解空间进行充分搜索，同样易于陷入局部极小值。Narendhar 等人[8]将 BFO 算法与蚁群算法相结合，提出了一种混合的 BFO 求解车间作业调度问题，但搜索能力随着搜索空间维数的增加迅速下降。并且由于 BFO 算法本身没有充分显示细菌菌落的优化优势，这些离散 BFO 算法在很大程度上与离散的微粒群算法类似。

由于 BCO 算法模拟了整个细菌菌落的演化过程，具有更为合理的进化机制，为此，本文将根据 BCO 算法的基本原理，提出一种混合的离散细菌菌落算法 HDBCO(hybrid discrete bacterial colony optimization algorithm)。该算法：a)针对组合优化问题，设计新的个体编码方式及进化模式；b)为了提高种群的活力，根据个体相似度提出一种自适应学习策略，帮助细菌个体跳出局部最优；c)在算法中融入禁忌搜索方法，以增强算法局部搜索能力；d)采用 Taillard 标准调度测试问题集在 MATLAB 软件中进行算法仿真测试。

**收稿日期**：2013-05-06；**修回日期**：2013-07-08　**基金项目**：国家自然科学基金资助项目(31100424)；云南省自然科学基金资助项目(2009CD070)；云南省教育厅科学研究基金资助项目(2012J047)

**作者简介**：宋德逻(1987-)，男，江西宜春人，硕士研究生，主要研究方向为智能控制、群集智能优化算法；孔德福(1987-)，男，安徽滁州人，硕士研究生，主要研究方向为智能控制、群集智能优化算法；李明(1977-)，男(通信作者)，江苏盐城人，副教授，博士，主要研究方向为智能控制、群集智能优化算法(lm_2001@163.com).

## 1　混合的离散细菌菌落优化算法

BCO 算法是李明等人于2011 年提出的一种模拟了细菌菌落生长演化过程的智能群集优化算法,该算法提出了新的个体进化机制及运动方式,同时提出了一种自然的结束准则。在 BCO 算法中每个细菌个体就是问题的一个可行解,同时制定细菌个体年龄,个体通过涌动、翻滚、停留等方式在解空间中进行搜索。当个体连续沿正的浓度方向的移动次数达到一定时,细菌繁殖;反之当个体连续沿负浓度方向的移动次数达到一定次数或者细菌沿正方向与负方向反复移动达到一定次数时,细菌死亡。由于细菌的繁殖速度快,因此设置了种群最大规模数 $S$。现有 BCO 算法主要用于解决连续函数优化问题,由于本文重点研究离散细菌优化算法,仅简单介绍 BCO 算法的基本原理,具体实现过程不再赘述。

由于组合优化问题的解空间是离散的,在利用细菌算法解决组合优化问题时,必须设计合适的个体表达方式。在文献[6~8]中采用了 LOV 编码规则,该规则根据个体位置分量在连续空间中的大小进行排序,并将排序后的序列视为问题的一个可行解,因此算法本质上还是在连续空间搜索最优解。由于算法搜索空间和实际排序问题的离散解空间之间不存在严格的对应关系,个体在连续空间中的优劣无法通过 LOV 编码直接反映在排序问题的解空间中,显然这些算法还是利用连续函数优化方法解决排序问题,不可避免地存在本质上的不足。

本文提出一种直接将工件的排列作为细菌位置矢量的编码方式,去除中间的编码转换,使得细菌可以直接在排序问题的解空间进行优化搜索。另外,根据群集优化算法基本原理,个体会向群体或个体历史最优位置移动,在连续优化空间中,可以通过简单的向量加减来实现,但无法直接运用到离散空间中。因此,本文需要对离散个体的这种移动方式重新定义。

a)在连续解空间中细菌具有沿特定方向移动的性质。为了在离散空间中也能体现这种方向性,本文针对两个位置向量 $A$、$B$ 定义了细菌进化速度 $v=(1,2,\cdots,n-1,n)$。由于在排序问题中,位置向量 $A$、$B$ 具有相同的分量只是位置不同,所以 $A$ 到 $B$ 的速度分量可以由 $A$ 中分量在 $B$ 中的排列位置来决定。例如,向量 $A=(a,c,b,d,e,f)$,$B=(b,c,d,e,a,f)$;向量 $A$ 中第一个分量 $a$ 是向量 $B$ 中的第五个分量,因此速度 $v$ 的第一个位置分量是5,向量 $A$ 中第二个分量 $c$ 是向量 $B$ 中的第二个分量,因此速度 $v$ 的第二个位置分量是2,依此类推,可得速度 $v_{A\to B}=(5,2,1,3,4,6)$。同理,$B$ 到 $A$ 的速度 $v_{A\to B}=(3,2,4,5,1,6)$。

b)为了使得细菌位置在离散域空间得以更新,实现在解空间的优化搜索,定义细菌的移动方式 $A\oplus B$,表示在 $A$ 位置上的细菌以速度 $v$ 向 $B$ 位置移动,新位置 $A'$ 各分量的值是由 $A$ 按照速度 $v_{A\to B}$ 中各分量的值对 $A$ 中各分量重新排列来确定。例如,上例中 $v_{A\to B}=(5,2,1,3,4,6)$,第一个位置分量是5,通过 $A\oplus B$ 移动,在 $A$ 中找出第 5 个位置分量的值 $e$,作为新位置 $A'$ 中的第一个位置分量。依此类推,依次找出新位置其他分量的值,可得新位置向量 $A'=A\oplus B=(e,c,a,b,d,f)$。

c)群集优化算法通常会加入随机因素以增强细菌的搜索能力。同样在离散域中为了体现这一思想,加入随机调整因子 $r$ 表示随机选择向量中的任意分量进行交换的次数。例如,向量 $A=(a,c,b,d,e,f)$,当 $r=2$ 时,表示随机两次交换向量中的任意分量,如随机交换向量中的 $a$ 与 $b$ 和 $c$ 与 $d$,得到新向量 $A'=(b,d,a,c,e,f)$。上述定义是本文直接针对离散解空间而提出的,现有文献之所以不用以上定义,是因为它们本质上还是在连续空间搜索。

基于以上定义,细菌个体进化过程可以表述为:当个体第 $k$ 次迭代的适应值优于第 $k-1$ 次时,认为细菌是一次成功的搜索,则个体在 $k+1$ 次迭代时选择涌动的搜索方式,快速地向营养物质高的区域移动;一旦个体第 $k$ 次迭代的适应值劣于第 $k-1$次时,则个体在第 $k+1$ 次时选择翻滚的搜索方式,细菌在原地随机搜索,探寻周围空间营养浓度高的区域。细菌进化模型如下:

$$\text{涌动}\quad x_{k+1}=r_1\otimes((x_k\oplus p_k)\oplus g_k) \tag{1}$$

$$\text{翻滚}\quad x_{k+1}=r_3\otimes x_k \tag{2}$$

其中:任意细菌的位置向量 $x_k=(x_1,x_2,\cdots,x_{n-1},x_n)$ 代表一个可行解;$p_k=(x_1',x_2',\cdots,x_{n'-1},x_n')$ 代表第 $k$ 个细菌所记住的历史最优位置;$g=(x''_1,x''_2,\cdots,x''_{n-1},x''_n)$ 代表种群最优位置。$r_1$、$r_3$ 是调整因子;$x_k\oplus p_k$ 表示细菌以速度 $v$ 向个体历史最优位置移动;$(x_k\oplus p_k)\oplus g_k$ 表示细菌结合历史信息以速度 $v$ 向种群最优位置移动;$r\otimes B$ 表示随机选择向量中的任意分量交换 $r$ 次。为了增加种群多样性,细菌每次移动前都对速度 $v$ 进行适当的调整。另外,与基本的 BCO 算法一样,本文提供两种结束方式:迭代次数;当细菌种群全部死亡之后,算法自然结束。

在迭代的过程中,细菌不断地向种群最优位置移动。然而当细菌位置接近种群最优位置 $g_{\text{best}}$ 时,其自身经历的最优位置 $p_{\text{best}}$ 也可能接近 $g_{\text{best}}$,由此细菌可能会较早地聚集于某一个位置,从而细菌个体位置很难更新。此时,算法引入一个相似度因子 $w$ 来判断细菌活力,其通过比较两个细菌个体中相同位置的分量是否相同来确定。例如上例中 $A=(a,c,b,d,e,f)$,$B=(b,c,d,e,a,f)$,由于两个向量中第 2、第 6 个元素均是 $c$、$f$,由此可得其相似度因子 $w=2$。当细菌出现停滞时,本文通过在翻滚时设置自适应调整因子 $r_3$ 来增强细菌活力,其更新原则如下:

$$r_2=c_1\times w_1+c_2\times w_2-m \tag{3}$$

$$r_3=p_1\quad r_2<0 \tag{4}$$

$$r_3=p_2^{r_2}\quad r_2\geqslant 0,\text{fix}(r_2) \tag{5}$$

其中:$w_1$ 代表当前细菌与其历史最优位置细菌相似程度;$w_2$ 代表当前细菌与种群中最优位置细菌相似程度;$m$ 是 $r_3$ 的阈值;$p$、$c_1$、$c_2$ 为常数。当 $r_2$ 是负数时,说明此时的细菌相似度较小,细菌拥有较强的活力,可以令 $r_3$ 为一个较小的常数 $p_1$,使得细菌快速向营养浓度高的位置移动;当 $r_2$ 是非负数时,说明此时的细菌相似度较大,认为此时的细菌缺乏活力,此时对 $r_2$ 取整操作后进行式(5)操作,调整因子 $r_3$ 会呈指数关系增大,快速地打乱细菌位置,增强群体的多样性。但是翻滚时 $r$ 的增大同时会令算法更具随机性,所以本文令涌动时调整因子为一个较小的常数,使细菌在涌动时主要进行细致搜索。

当种群连续迭代一定次数其最优值不变时,此时认为细菌陷入了早熟,本文通过融入禁忌搜索算法,制作禁忌表、设定禁忌长度和藐视规则,使得细菌进行精确搜索,跳出局部极值。细菌在禁忌搜索算法中进行迭代一定次数后,跳出禁忌搜索重新进入离散细菌菌落算法进行全局搜索;同时保留禁忌表避免下次进入禁忌算法时做不必要的重复工作。如此反复直到满足终止条件、结束算法。

HDBCO 算法步骤如下：

a)初始化。初始细菌位置，设定初始种群 $n$、最大种群规模 $S$、最大重复次数 max_iter、个体最长寿命 $N$、繁殖条件 $N_H$、系数迭代次数 $K_1$、$K_2$、禁忌长度 $L_2$、系数 $p_1$、$c_1$、$c_2$、$r_1$、$r_2$、$p_2$、$m$ 等。

b)评价每个细菌的适应度值，即计算每个细菌的目标函数值。

c)根据细菌目标函数值的优劣，计算细菌年龄，对细菌分类。

d)对正常细菌操作，计算细菌的相似度 $w$，以及自适应调整因子 $r_3$；获取细菌进化速度 $v$，并且对速度适当调整之后，对每个细菌分别按照进化模型进行操作，更新细菌位置；评价每个细菌的目标函数值，更新 $p_{best}$；淘汰达到死亡条件的细菌。

e)繁殖达到繁殖条件的细菌，并评价每个细菌的目标函数值，更新 $g_{best}$。

f)判断 $g_{best}$ 是否达到 max_iter。若是，则进入禁忌搜索算法，在禁忌搜索算法中搜索一定次数 $K_2$ 之后跳出禁忌搜索算法，更新 $g_{best}$；否则转 b)。

g)若满足终止条件，停止算法，否则转 b)。

## 2 仿真实例

为了客观评价 HDBCO 算法的性能，本文采用 MATLAB 软件对其仿真，并且选取 Taillard 调度测试问题集对算法的性能进行测试，分别与 SHOP[10] 算法、MMAS[10] 算法、HDCPSO 算法[11]、NPSO [12]算法在标准调度测试问题集的性能进行比较。

HDBCO 参数设为：初始个体 10，最大种群数 $S=30$，个体最长寿命 $N=6$，繁殖条件 $N_H=2$，系数 $p_2=2$，$c_1=0.325$，$c_2=0.40$，最大重复数 max_iter = 80，迭代次数 $K_1=500$，$K_2=30$。当 jobs = 20 时，$r_1=r_2=1$，$p_1=2$，$m=12$，$L_2=12$；当 jobs = 50 时，$r_1=r_2=2$，$p_1=2$，$m=28$，$L_2=8$；当 jobs = 100 时，$r_1=r_2=2$，$p_1=3$，$m=67$，$L_2=8$。

表 1 中列出了从 20/5 ~ 100/20 的九种不同维数测试问题集，其中每一维有 10 个不同的问题实例，每一个问题实例测试 20 次。表 1 中列出的计算结果是相对于流水作业调度问题标准测试库中给出的最好结果的平均百分比偏差 $\xi$。

$$\xi = 100\left(\frac{\bar{x}-\mu}{\mu}\right) \tag{5}$$

其中：$\bar{x}$ 为 HDBCO 算法中所得最好结果的平均值，$\mu$ 为测试库中给出的最好结果。

表 1 本文算法与其他算法的性能比较

| N/M | 平均偏差($\xi$)/% | | | | |
|---|---|---|---|---|---|
| | SHOP | HDCPSO | NPSO | MMAS | HDBCO |
| 20/5 | 1.061 | 0.00 | 0.963 | 0.408 | 0.787 |
| 20/10 | 1.462 | 0.327 | 2.173 | 0.591 | 0.755 |
| 20/20 | 1.116 | 0.217 | 1.50 | 0.410 | 0.409 |
| 50/5 | 0.597 | 0.003 | 0.093 | 0.145 | 0.510 |
| 50/10 | 3.012 | 1.52 | 3.883 | 2.193 | 1.693 |
| 50/20 | 3.147 | 1.92 | 4.62 | 2.475 | 2.418 |
| 100/5 | 0.509 | 0.009 | 0.19 | 0.196 | 0.534 |
| 100/10 | 1.719 | 0.533 | 1.286 | 0.928 | 1.891 |
| 100/20 | 3.249 | 2.14 | 4.193 | 2.238 | 2.509 |

从表 1 中可以看出，HDBCO 算法的总体搜索能力与 NPSO 算法相当，稍劣于 HDCPSO 算法；但随着问题规模的增加，无论是在搜索精度还是稳定性方面 HDBCO 算法都强于 SHOP 算法和 MMAS 算法。另外，本文算法在 20/5 ~ 50/20 的六种不同维数测试问题集的每一个问题实例中都能找到标准测试问题集中列出的最优解；特别是在 20/10 和 20/20 种子初始值等于 268827376 时，标准测试问题集中其最优解分别为 1378 和 2100。本文的 HDBCO 算法寻求到了优于测试问题集中所给出的最优解，其最优解分别为 1377 与 2099，最优解排列分别为[3,11,6,20,10,12,16,18,13,7,1,9,2,15,14,19,8,4,5,17]与[18,11,6,12,4,15,1,16,19,13,8,14,3,10,20,5,9,7,17,2]。仿真结果表明，HDBCO 算法具有平衡全局寻优以及局部精确搜索的能力，是一种能有效解决组合优化问题的群集智能优化算法。

## 3 结束语

本文在基本 BCO 算法中通过引入一种直接的解的编码方式和制定离散解空间的进化准则，提出了一种混合的离散细菌菌落优化算法。该算法通过设计相似度因子来判断种群活力，使得算法能够自适应改变调整因子的值以增加种群多样性，并且当算法陷入早熟收敛时融入禁忌搜索算法，使得算法能够跳出局部极值。仿真实验表明，该算法能够有效解决组合优化问题，并且更新了 Taillard 调度测试问题集中最优解，进一步表明了算法具备良好的稳定性及精确搜索能力。但是，算法中参数的选择对算法影响很大，参数的设置还需进一步探讨。

**参考文献：**


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

[2] MISHRA S, BHENDE C N, LAI L L. Optimization of a distribution static compensator by bacterial foraging technique[C]//Proc of International Conference on Machine Learning and Cybernetics. Washington DC: IEEE Computer Society, 2006: 4075-5082.

[3] ULAGAMMAI L, VANKATESH P, KANNAN P S, *et al*. Application of bacteria foraging technique trained and artificial and wavelet neural networks in load forecasting[J]. Neurocomputing, 2007, 70(16-18): 2659-2667.

[4] DANG Jing, BRABAZON A, O'NEILL M, *et al*. Optionmodel calibration using a bacterial foraging optimization algorithm[C]//Proc of Conference on Applications of Evolutionary Computing. Berlin: Springer-Verlag, 2008: 113-122.

[5] 李明，杨成梧. 细菌菌落优化算法[J]. 控制理论与应用，2011，28(2)：223-228.

[6] 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

[7] 张娜. 细菌觅食优化算法求解车间调度问题的研究[D]. 长春：吉林大学，2007.

[8] NARENDHAR S, AMUDHA T. A hybrid bacterial foraging algorithm for solving Job-Shop scheduling problems[J]. International Journal of Programming Languages and Applications, 2012, 2(4): 1-7

[9] TAILLARD E. Benchmarks for basic scheduling problems[J]. European Journal of Operational Research, 1993, 64(2): 278-285.

[10] 王笑蓉，吴铁军. Flow-Shop 问题的蚁群优化调度方法[J]. 系统工程理论与实践，2003，23(5)：65-71.

[11] 田野，刘大有. 求解流水车间调度问题的混合粒子群算法[J]. 电子学报，2011，39(5)：1087-1093.

[12] LIAN Zhi-gang, GU Xing-sheng, JIAO Bin. A novel particle swarm optimazation algorithm for permutation Flow-Shop scheduling to minimize makespan[J]. Chaos, Solitons & Fractals, 2008, 35(5): 851-861.