Journal of Computer Applications
计算机应用,2012,32(2):428-431
ISSN 1001-9081
CODEN JYIIDU
2012-02-01
http://www.joca.cn

文章编号:1001-9081(2012)02-0428-04
doi:10.3724/SP.J.1087.2012.00428

# 基于微粒群与混合蛙跳融合的群体智能算法

孙 辉[1],龙 腾[2*],赵 嘉[1]
(1.南昌工程学院 信息工程学院,南昌 330099; 2.南昌航空大学 信息工程学院,南昌 330063)
(*通信作者电子邮箱 longteng5740@163.com)

**摘 要**:针对微粒群算法和混合蛙跳算法存在的早熟收敛问题,提出一种基于微粒群与混合蛙跳算法融合的群体智能算法。新算法将整个群体分成数目相等的蛙群和微粒群群体。在两群体独立进化过程中,设计了一种两群之间的信息替换策略:比较蛙群与微粒群的最佳适应值,如果蛙群进化较好,利用蛙群各子群中最差个体替换微粒群一部分较好个体;否则,用微粒群中较好的一部分个体替换蛙群各子群的最好个体。同时,设计了一种两群之间的相互协作方式。为避免微粒群因早熟收敛而影响信息替换策略效果,适时对其所有个体最好位置进行随机扰动。仿真实验表明,新算法可以有效提高全局搜索能力及收敛速度,对于高维复杂函数问题,算法具有很好的稳定性。

**关键词**:微粒群算法;混合蛙跳算法;信息替换策略;随机扰动;协作方式
**中图分类号**:TP183 **文献标志码**:A

# Swarm intelligence algorithm based on combination of shuffled frog leaping algorithm and particle swarm optimization

SUN Hui[1], LONG Teng[2*], ZHAO Jia[1]
(1. *School of Information Engineering, Nanchang Institute of Technology, Nanchang Jiangxi* 330099, *China*;
2. *School of Information Engineering, Nanchang Hangkong University, Nanchang Jiangxi* 330063, *China*)

**Abstract:** Concerning the premature convergence of Particle Swarm Optimization (PSO) algorithm and Shuffled Frog Leaping Algorithm (SFLA), this paper proposed a swarm intelligence optimization algorithm based on the combination of SFLA and PSO. In this algorithm, the whole particle was divided into two equal groups: SFLA and PSO. An information replacement strategy was designed in the process of their iteration: comparing the fitness of PSO with that of SFLA, the worst individual in each subgroup of SFLA would replace some better individuals in PSO when SFLA is better; otherwise, some better individuals in PSO would replace the best individual in each subgroup of SFLA. Meanwhile, a collaborative approach between the two groups was also designed. Since the information replacement strategy could be influenced by the premature convergence problem in PSO, a random disturbance would be given on each particle's best position. The simulation results show that the proposed algorithm can improve the global search ability and convergence speed efficiently. For the complex functions with high-dimension, the algorithm has very good stability.

**Key words:** Particle Swarm Optimization (PSO); Shuffled Frog Leaping Algorithm (SFLA); information replacement strategy; random disturbance; collaborative approach

## 0 引言

自 Bergh 等[1]提出协同进化微粒群算法(Cooperative Particle Swarm Optimization, CPSO)算法以来,已有很多学者提出了其他智能算法与微粒群的混合进化方法[2-6]。然而,混合蛙跳(Shuffled Frog Leaping Algorithm,SFLA)[7]与微粒群算法(Particle Swarm Optimization, PSO)[8]的融合方法研究近年才开始。Niknam 等[9]提出改进自适应微粒群算法(self-adaptive particle swarm optimization,SAPSO),并将该算法与蛙跳算法进行融合提出 SAPSO-SFLA 算法:将 $3N$($N$ 为维数)个个体按适应值排序,较好的前 $N$ 个个体按 SFLA 进行进化,其他 $2N$ 个个体则按照 SAPSO 算法进行进化。然后将得到的 $3N$ 个微粒按适应值重新排序,重复上述步骤至满足条件结束。张尤赛等[10]提出 SFLA-PSO 算法,利用 SFLA 算法的多种群的进化方式进行族群的混选,然后利用 PSO 算法进行族群局部搜索。潘玉霞等[11]提出微粒群—蛙跳协同算法(P-SC)算法:把整个群体分成数目相等的蛙群和微粒群群体,两个群体共享整个群体搜索到的最优解来更新微粒位置。以上算法都不同程度结合 PSO 算法速度快,SFLA 通用性强、局部搜索能力较强等特点,搜索最优解能力得到不同程度的改善。但以上算法 PSO 群与 SFLA 群之间未得到充分的信息替换和共享,融合不够充分,协作方式不够彻底。

鉴于此,本文提出一种新的微粒群与蛙跳的融合算法,称为 PS_LT。该算法通过相互协作方式,共同提升两种群的进化,共同追寻最优解。为了克服两群体陷入局部最优的可能性,建立一种两群体之间的信息替换策略;同时,适时对微粒群所有个体最好位置进行随机扰动,保持信息替换策略的有效性。实验表明,该算法可以有效提高算法的全局搜索最优解能力、收敛速度以及在高维时具有稳定的寻优性能。

收稿日期:2011-07-18;修回日期:2011-09-20。
基金项目:国家自然科学基金资助项目(61162022);江西省自然科学基金资助项目(2010GZS0163,2009GZS0083)。
作者简介:孙辉(1959-),男,江西九江人,教授,博士,主要研究方向:计算智能、变分不等原理及变分不等式、多尺度几何分析、图像处理;龙腾(1986-),男,江西南昌人,硕士研究生,主要研究方向:群智能算法; 赵嘉(1981-),男,江西九江人,讲师,硕士,主要研究方向:群智能算法。

# 1　微粒群算法与混合蛙跳算法原理

## 1.1　微粒群算法

微粒群算法描述了 $n$ 个微粒在 $d$ 维搜索空间以一定速度飞行情况。微粒具有位置和速度两个属性。第 $i$ 个微粒位置表示为 $X_i = (x_{i1}, x_{i2}, \cdots, x_{id})$，速度表示为 $V_i = (v_{i1}, v_{i2}, \cdots, v_{id})$。算法通过适应值函数来衡量微粒的位置好坏，通过不断迭代来寻求最优位置。迭代过程中，微粒不断根据自己的飞行经验和同伴经验来决定移动到下一位置，历史经验即为个体最好位置，同伴经验即为所有粒子发现的最好位置。其进化公式如下:

$$V_{ij}(k+1) = wV_{ij}(k) + c_1 r_1 (p_{ij} - x_{ij}(k)) + c_2 r_2 (p_{gj} - x_{ij}(k)) \quad (1)$$

$$x_{ij}(k+1) = x_{ij}(k) + v_{ij}(k+1) \quad (2)$$

其中: $i = 1, 2, \cdots, n$, $j = 1, 2, \cdots, d$; $c_1$、$c_2$ 为加速因子; $w$ 为惯性权重; $r_1$、$r_2$ 是相互独立的随机数，服从 $(0,1)$ 上的均匀分布; $k$ 为当前的迭代次数; $p_{ij}$ 为个体最好位置; $p_{gj}$ 为全局最好位置。本文所提算法中，运用式(3)进行速度更新:

$$V_{ij}(k+1) = 0.5(wV_{ij}(k) + c_1 r_1 (p_{ij} - x_{ij}(k)) + c_2 r_2 (p_{gj} - x_{ij}(k))) \quad (3)$$

微粒群算法具有收敛速度较快，设置参数少，但是优化过程中存在容易陷入局部最优的缺陷。

## 1.2　混合蛙跳算法

混合蛙跳算法模拟了 $P$ 只青蛙在 $d$ 维搜索空间不断觅食过程。随机初始化蛙跳位置，按适应值排序，将整个青蛙群体划分 $F$ 个子群，每个子群里面有 $G$ 只青蛙，满足 $P = F * G$; 每个子群的青蛙个体都有自己的思想，它们的思想因为交流而相互影响。划分好子群后，每个子群都按照各自群文化进行局部搜索，得到子族群的最好青蛙和最差青蛙; 全群最好青蛙 $PX$ 为子族群最好青蛙的最好个体。局部搜索的目的就是要在规定迭代次数之内，改善最差青蛙位置。局部搜索按式(4)、(5)进行改善:

$$D_i = \text{rand}() * (P_b - P_w) \quad (4)$$

$$P_{new\_w} = P_w + D_i;\ D_{min} \leqslant D_i \leqslant D_{max} \quad (5)$$

其中: $D_i$ 表示最差青蛙在第 $i$ 维移动的距离，$P_w$ 为最差青蛙位置，$P_b$ 为子族群中最好青蛙位置，$P_{new\_w}$ 为青蛙改善后位置，$D_{min}$、$D_{max}$ 表示移动的最小距离和最大距离，$\text{rand}() \in [0,1]$ 的随机数。如按式(4)和(5)不能改善最差青蛙，则用 $PX$ 位置代替式(4)中最好青蛙位置。如再不能得到改善，则随机产生一个解。

局部搜索之后，各个子群重新合成新一代种群并排序和划分子族群，然后各个子族群重新执行局部搜索，重复此过程直至满足条件结束。全局的信息交换和局部的深度搜索使得青蛙信息得到充分的交流，也使得 SFLA 算法有较强的跳出局部极值能力，但算法也存在早熟收敛和收敛速度较慢等缺陷。

# 2　基于 PSO 与 SFLA 的融合群体智能算法

本文提出的算法主要通过两群体之间的协作来达到提升两个群体之间的进化，其协作方式为用一个进化较好的群体信息提升较差群体，让较差群体得到改善，这种协作方式贯穿于整个迭代过程。为了增强较差群体多样性，设计一种信息替换策略，其能够使得两群体之间得到充分的信息替换和共享，让较优好群体的微粒去指导较差群体的优化。将此算法命名为 PS_LT。

## 2.1　PS_LT 协作方式

将整个群体分成数目相等的微粒群与蛙群群体，分别独立迭代一次，如果微粒群进化所得最佳适应值 $GX$ 较好，认为此时微粒群进化较好，然后执行信息替换策略，以提高蛙群种群多样性，再转到蛙群进化过程，提升蛙群进化; 如果蛙群进化所得最佳适应值 $GY$ 较好，认为此时蛙群进化较好，然后执行信息替换策略，以提高微粒群多样性，再转到微粒群进化过程，提升微粒群进化。重复此过程至满足迭代条件结束。这样两群体进化都能够得到提升，共同追寻最优解。

## 2.2　信息替换策略

两群体每迭代一次，保存微粒群中前 $m$($m$ 数值等于蛙群中子族群数目)个较好微粒和蛙群中每个子族群的最好微粒与最差微粒，并将保存的这些微粒进行编号，编号方式为: 第 $m$ 个微粒标号保存为 $PB[1]$，第 $m-1$ 个微粒标为 $PB[2]$，第 $j$ 个微粒保存为 $PB[m-j+1]$，以此类推，第 1 个微粒标号为 $PB[m]$。将蛙群中第 1 个子族群的最好个体保存为 $Pb[1]$、第 2 个子族群最好个体保存为 $Pb[2]$，第 $j$ 个子族群最好个体保存为 $Pb[j]$，以此类推，第 $m$ 个子族群的最好个体保存为 $Pb[m]$; 相应的，蛙群中第 $j$ 个子族群最差个体保存为 $Pw[j]$，其中 $j$ 取值从 1 至 $m$。以下的替换策略都将围绕这些保存的这些微粒进行。当微粒群每迭代一次得到的最佳适应值 $GX$ 优于蛙群最佳适应值 $GY$，进一步依次比较 $PB[m-j+1]$ 与 $Pb[j]$ 适应值，如 $PB[m-j+1]$ 适应值较好，则将 $PB[m-j+1]$ 个体位置替换 $Pb[j]$ 个体位置; 否则，不替换。目的是让微粒群得到的较好个体去引导蛙群中每个子族群的进化，增强青蛙个体活性，提高蛙群子种群多样性。

因为微粒群进化不仅跟全局最好位置有关，其全局最差位置可以扩大种群的搜索范围。因此，可以利用蛙群中每个子族群的最差青蛙 $Pb[j]$ 来扩大微粒群的搜索范围，同时增强微粒群个体活性和全群多样性。即当每迭代一次所得蛙群最佳适应值 $GY$ 优于微粒群最佳适应值 $GX$ 时，进一步依次比较 $PB[m-j+1]$ 与 $Pw[j]$ 个体适应值，如 $Pw[j]$ 个体适应值较差，将蛙群每个子族群的最差青蛙 $Pw[j]$ 个体位置替换微粒群的 $PB[m-j+1]$ 个体位置。

这样通过不断将微粒群中一部分较好个体与蛙群中子族群最差个体或者最好个体进行替换，使得两群体之间的信息得到不断交流。当 PSO 进化较好，通过将 PSO 微粒替换 SFLA 微粒来提升 SFLA 进化; 当 SFLA 进化较好，则将 SFLA 微粒替换 PSO 微粒，以提升 PSO 进化，然后循环以上过程直至满足迭代条件结束，这样蛙群和微粒群相互促进，共同提升地去追寻最优解。

同时，因为微粒群极易快速收敛到局部最优解，微粒失去活性，那么这时基本就是具有同样属性的粒子去替换蛙群中的微粒，这样替换效果会不太理想。因此，假设微粒群全局最好值在规定次数内如不能得到改善，则对 PSO 所有个体最好位置在原有位置上进行随机扰动。扰动公式如下:

$$PB[j].position[k] = PB[j].position[k] + \text{rand}() * PB[j].position[k] \quad (6)$$

其中: $\text{rand}()$ 是属于 $[0,1]$ 的随机数，$PB[j].position[k]$ 为微粒群第 $j$ 个个体最好位置。

## 2.3　算法流程

PS_LT 算法流程如下:

步骤 1　初始化微粒群与蛙群参数，并记下微粒群全局最好值和个体最好值、蛙群每个子族群中最好青蛙和最差青蛙以及蛙群全群最好青蛙。

步骤 2　令 $w$ 从 1.8 线性减小到 0.9。

步骤 3　对蛙群适应值进行排序并将其分为 $m$ 个子族群，并对微粒群个体极值进行排序，选出前 $m$ 个最好个体保存到 $PB[j]$; 同时将蛙群中子族群最好个体和最差个体进行分别保存到 $Pb[j]$ 和 $Pw[j]$。

步骤4 判断 $GX$ 与 $GY$ 的大小。如果 $GX$ 较好，进一步将微粒群 $PB[m-j+1]$ 适应值与蛙群 $Pb[j]$ 适应值依次进行比较，如果 $PB[m-j+1]$ 适应值较好，则将 $PB[m-j+1]$ 位置替换 $Pb[j]$ 位置；否则，不替换。然后按照式(4)(5)执行SFLA局部搜索。如果 $GY$ 较好，进一步将蛙群 $Pw[j]$ 适应值依次与微粒群中 $PB[j]$ 适应值进行比较，如 $Pw[j]$ 适应值较差，则将蛙群中 $Pw[j]$ 的位置替换微粒群中 $PB[m-j+1]$ 位置；否则，不替换。然后按照式(2)(3)执行PSO搜索。

步骤5 判断微粒群的最好值 $GX$ 不变次数是否达到规定值，如达到则对微粒群的个体极值按式(6)进行随机扰动；否则，不扰动。

步骤6 输出两群最优解的较好解作为结果，判断迭代终止条件，如达到，则结束；否则，转向步骤2。

## 3 仿真实验

本文采用几个经典测试函数对上述所提两种策略进行了实验，采用C语言进行编程。测试函数如表1。

表1 测试函数

| 函数 | 搜索空间 | 理论最优值 | 迭代次数 |
|---|---|---|---|
| Sphere | [-100,100] | 0 | 2000 |
| Griewank | [-600,600] | 0 | 3000 |
| Rastrigrin | [-5.12,5.12] | 0 | 3000 |
| Ackley | [-32,32] | 0 | 3000 |

以上函数理论值0都会在 $(x_1,x_2,\cdots,x_D)=(0,0,\cdots0)$ 处取得，其中 $D$ 表示解的维数。

算法PS_LT参数：PSO群数目和蛙群数目均为200，蛙群含有20个子族群，每个子族群有10只青蛙。微粒群加速因子均为2.0，惯性权重 $w$ 从1.8线性递减到0.9。蛙群最大移动步长大致取各函数最大搜索范围的1/6。微粒群最好值最大不变次数设为12。为了确定PS_LT算法中蛙群各子群内部迭代次数 $S\_N$ 最佳值，做了Rastrigrin函数平均最优适应值 $Aver$ 随 $S\_N$ 变化情况实验，进行100次独立实验。如表2所示，根据表2数据本文将 $S\_N$ 设为30。

表2 Rastrigrin函数平均适应值随 $S\_N$ 变化情况

| $S\_N$ | $Aver$ | $S\_N$ | $Aver$ |
|---|---|---|---|
| 10 | 4.615156 | 40 | 1.220232 |
| 20 | 4.352898 | 60 | 2.211693 |
| 30 | 0.907240 | 100 | 9.055068 |

表3为100次独立实验，PS_LT与基本SFLA、惯性权重线性递减微粒群算法(Linearly Decreasing Weight Particle Swarm Optimization，LDWPSO)[12]、文献[10]中的SFLA-PSO算法得到的平均最优适应值($Aver$)和最好适应值($Goodfit$)情况。

从表3中可以看出，PS_LT除了在Rastrigin函数40维上平均值未得到较大改善外，对其他测试函数表现出较好的寻优最优解能力，在寻优到最好解方面都具有较大优势。总体来说，相比于基本蛙跳算法(SFLA)、线性递减惯性权重微粒群算法(LDWPSO)和已有SFLA-PSO混合策略，本文算法(PS_LT)的寻优性能有较大提高。

为了测试PS_LT算法收敛速度，图1给出了上述各种算法对20维Griewank函数测试结果，迭代次数2000。从图中可以看出PS_LT与SFLA收敛速度较其他两种算法较快，而本文测得PS_LT算法在602代已收敛于全局最优解4.537860E-03，而SFLA算法在817代才趋于最优解0。因此，PS_LT具有较快的收敛速度。

表3 算法的性能比较

| 函数 | 维数 | LDWPSO | | SFLA | |
|---|---|---|---|---|---|
| | | $Aver$ | $Goodfit$ | $Aver$ | $Goodfit$ |
| Sphere | 50 | 1.012292E-08 | 3.516598E-09 | 6.930908E-10 | 3.103205E-12 |
| | 80 | 5.443501E-04 | 1.239704E-04 | 1.364702E-07 | 2.493595E-07 |
| | 100 | 2.365715E-02 | 6.732474E-03 | 5.397777E-04 | 1.243252E-05 |
| Griewank | 20 | 2.969529E-02 | 0.000000E+00 | 4.430020E-02 | 0.000000E+00 |
| | 40 | 1.568145E-02 | 1.563194E-13 | 1.337863E-02 | 1.732278E-19 |
| | 50 | 3.144116E-02 | 1.504951E-07 | 9.181977E-03 | 4.498796E-12 |
| Rastrigrin | 10 | 1.342811E-01 | 5.728693E-03 | 1.616495E-12 | 1.248727E-17 |
| | 20 | 1.470907E+00 | 1.457805E-01 | 1.990655E-01 | 4.162278E-12 |
| | 40 | 6.249902E+00 | 7.358354E-01 | 8.041537E+00 | 1.989914E+00 |
| Ackley | 20 | 4.080292E-10 | 4.130030E-12 | 1.831513E-08 | 5.047474E-12 |
| | 40 | 6.477298E-07 | 3.124686E-08 | 1.034290E-03 | 4.786169E-06 |
| | 50 | 2.432934E-04 | 2.296733E-09 | 1.165906E-01 | 7.683967E-05 |

| 函数 | 维数 | SFLA-PSO | | PS_LT | |
|---|---|---|---|---|---|
| | | $Aver$ | $Goodfit$ | $Aver$ | $Goodfit$ |
| Sphere | 50 | 4.658356E-05 | 1.513493E-05 | 4.414036E-14 | 6.678416E-15 |
| | 80 | 1.430313E-02 | 4.966696E-02 | 2.033860E-10 | 2.491431E-12 |
| | 100 | 3.263255E-01 | 2.202175E-01 | 1.247618E-08 | 2.211568E-09 |
| Griewank | 20 | 6.678535E-02 | 9.904196E-13 | 1.219512-002 | 0.000000E+00 |
| | 40 | 1.153573E-02 | 1.441280E-08 | 2.447949E-03 | 0.000000E+00 |
| | 50 | 2.612475E-02 | 2.007218E-07 | 3.440792E-03 | 0.000000E+00 |
| Rastrigrin | 10 | 1.075336E+00 | 9.093495E-07 | 0.000000E+00 | 0.000000E+00 |
| | 20 | 1.192301E+01 | 8.011745E+00 | 2.728816E-02 | 0.000000E+00 |
| | 40 | 3.789549E+01 | 3.107826E+01 | 5.334837E+00 | 9.772977E-01 |
| Ackley | 20 | 1.505546E-02 | 1.379687E-02 | 5.887218E-16 | 4.821230E-16 |
| | 40 | 2.163366E+00 | 2.113648E+00 | 2.358315E-06 | 5.887218E-16 |
| | 50 | 6.702246E+00 | 6.649247E+00 | 1.143518E-02 | 4.141435E-15 |

为了说明 PS_LT 算法在求解高维表现出的稳定性能,与文献 [9] 中 SAPSO-SFLA 算法及文献 [10] 中 SFLA-PSO 算法进行比较,在相同参数下,测试了以上三种算法求解 Griewank、Rastrigrin 函数在 80 维、100 维的平均适应值(20 次独立实验) 方差。如表 4 所示。

从表 4 中可以看出,在 80 维和 100 维 PS_LT 算法优化结果比较稳定,数据之间的波动性比较小,PS_LT 在高维还是表现出一定的稳定性能。

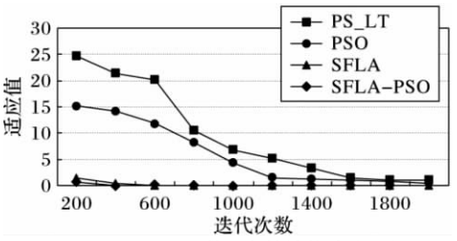


图 1　Griewank 函数下各算法的迭代进程

表 4　三种算法所求方差值

| 函数 | PS_LT | | SAPSO-SFLA | | SFLA-PSO | |
|---|---|---|---|---|---|---|
| | 80 维方差 | 100 维方差 | 80 维方差 | 100 维方差 | 80 维方差 | 100 维方差 |
| Griewank | 3.584 629 E-03 | 8.886 839 E-03 | 2.008 819 E-02 | 6.621 557 E-02 | 3.067 454 E-01 | 3.124 476 E+00 |
| Rastrigrin | 4.024 731 E+00 | 2.221 701 E+00 | 6.124 944 E+00 | 5.221 764 E+00 | 9.241 887 E+00 | 6.154 548 E+00 |

## 4　结语

针对微粒群算法和混合蛙跳算法特点,本文提出一种基于微粒群与混合蛙跳算法融合的群体智能算法。算法通过充分替换和共享两个独立群体的微粒信息,做到相互融合,共同提升去追寻最优解。仿真实验表明,新算法具有良好的全局搜索最优解性能,有效克服了微粒群和蛙群算法由于早熟收敛而导致的易陷入局部最优的缺陷; 同时,算法收敛速度得到有效提高,在求解高维时表现出较高的稳定性。

**参考文献:**


[1] BERGH F, ENGELBRECHT A P. A cooperative approach to particle swarm optimization[J]. IEEE Transactions on Evolutionary Computation, 2004, 8(3): 1-15.

[2] 王丽芳,曾建潮. 基于微粒群算法和模拟退火算法的协同进化方法[J]. 自动化学报,2006,32(4): 630-635.

[3] 陶新民,徐晶,杨立标,等. 改进的多种群协同进化微粒群优化算法[J]. 控制与决策. 2009, 24(9): 1406-1411.

[4] HAO ZHI-FENG, GUO GUANG-HAN, HUANG HAN. A particle swarm optimization algorithm with differential evolution [C]// International Conference on Machine Learning and Cybernetics. Piscataway: IEEE, 2007: 1030-1035.

[5] 王联国, 施秋红, 洪毅. PSO 和 AFSA 混合优化算法[J]. 计算机工程, 2010,36(5): 176-178.

[6] 王俊年,申群太,沈洪远,等. 基于多种群协同进化微粒群算法的径向基神经网络设计[J]. 控制理论与应用,2006,23(2): 251-255.

[7] EUSUFF, LANSEY K E. Optimization of water distribution network design using the shuffled frog leaping algorithm[J]. Journal of Water Resources Planning and Management,2003,129(3): 210-225.

[8] KENNEDY J, EBERHART R C. Particle swarm optimization [C]// Proceedings of IEEE International Conference on Neural Networks. Piscataway: IEEE, 1995: 1942-1948.

[9] NIKNAM T, FARSANI E A. A hybrid evolutionary algorithm for distribution feeder reconfiguration [J]. Science China: Technological Sciences,2010,53(4): 950-959.

[10] 张尤赛,高孟琦. 基于混洗蛙跳和粒子群优化算法的块自增纹理合成[J]. 计算机应用,2011,31(2): 366-368.

[11] 潘玉霞,谢光,潘全科. 批量无等待调度问题的微粒群蛙跳混合优化算法[J]. 计算机应用研究,2011, 28(2): 461-465.

[12] SHI Y, EBERHART R C. A modified particle swarm optimizer [C]// Proceedings of IEEE Congress Evolutionary Computation. Piscataway: IEEE, 1998: 69-73.


(上接第 427 页)

**参考文献:**


[1] APPLEGATE D L, BIXBY R E, CHVáTAL V, *et al*. The traveling salesman problem: A computational study (Princeton in applied mathematics) [M]. Princeton: Princeton University Press, 2007.

[2] MARTIN O C, OTTO S W, FELTEN E W. Large-step Markov chains for the traveling salesman problem [J]. Complex Systems, 1991, 5(3): 299-326.

[3] HELSGAUN K. An effective implementation of the Lin-Kernighan traveling salesman heuristic [J]. European Journal Operation Research, 2000, 126(1): 106-130.

[4] WALSHAW C. A multilevel Lin-Kernighan-Helsgaun algorithm for the travelling salesman problem, mathematics research report: 01/IM/80 [R]. London: University of Greenwich, Computing and Mathematical Sciences, 2001.

[5] 邹鹏, 周智, 陈国良, 等. 求解 TSP 问题的多级归约算法[J]. 软件学报,2003,14(1): 35-42.

[6] 邹鹏,周智,江贺,等. 求解旅行商问题的循环局部搜索算法的运行时间和性能分布分析[J]. 计算机学报, 2006, 29(1): 92-99.

[7] 王东, 吴湘滨. 提高链式 Lin-Kernighan 算法性能的策略[J]. 计算机应用,2007,27(11): 2826-2829.

[8] FISCHER T, MERZ P. Reducing the size of traveling salesman problem instances by fixing edges [C]// EvoCOP 07: Seventh European Conference on Evolutionary Computation in Combinatorial Optimisation, LNCS 4446. Berlin: Springer-Verlag, 2007: 72-83.

[9] 林冬梅, 王东, 钟勇. 确定 TSP 全局最优解部分边的蒙特卡罗模型[J]. 小型微型计算机系统. 2010,31(4): 747-751.

[10] BOESE D K. Cost versus distance in the traveling salesman problem, TR-950018 [R]. Los Angeles: University of California, Computer Science Department, 1995.

[11] University of Heidelberg. Traveling salesman problems library [EB/OL]. [2011-04-22]. http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/.

[12] DAVID A, ROBERT B, VASEK C. Concorde network optimization package [CP/OL]. [2011-04-15]. http://www.tsp.gatech.edu/concorde/downloads/codes/src/co031219.tgz.