文章编号：2095-2163(2019)01-0098-06 中图分类号：TP18 文献标志码：A

离散型鸡群优化算法在 0-1 背包中的应用

周 洋，潘大志
（西华师范大学 数学与信息学院，四川 南充 637000）

摘 要：为了进一步拓宽鸡群算法的研究领域，设计一种离散型鸡群算法（DCSO）。针对 0-1 背包问题的特点，在基本鸡群算法的基础上，对更新后的鸡群进行离散化处理，同时，在公鸡的位置更新过程中，引入自适应权重组合变异算子并动态调整变异权重，增强种群的多样性，更好地维持算法的“开采”与“搜索”两个阶段的平衡。最后，采用贪心修复算子对不可行解进行修正。通过 4 个经典 0-1 背包问题实例的仿真结果表明，相比离散粒子群算法、遗传算法和蚁群算法，DCSO 算法在解的质量、收敛速度以及鲁棒性等方面效果显著提升，验证了该算法的可行性。

关键词：鸡群算法；0-1 背包问题；离散化；自适应变异

The application of the discrete chicken swarm optimization（DCSO） in 0-1 knapsack problem

ZHOU Yang, PAN Dazhi
（College of Mathematic & Information, China West Normal University, Nanchong Sichuan 637000, China）

【Abstract】 In order to further expand the research field of chicken swarm optimization, a discrete chicken swarm optimization (DCSO) algorithm is designed. According to the characteristics of the 0-1 knapsack problems, based on the chicken swarm algorithm, this paper discretes the updated chickens. At the same time, in the process of the rooster location update, this algorithm employs the adaptive weight combined mutation operator and adjusts the weight of mutation dynamically, enhances the diversity of population, better balances the "development" and "exploration" of the algorithm.Finally, greedy repair operator is used to modify the unfeasible solutions. Experiments are conducted on the 4 classical 0-1 knapsack problems, all results show that compared with discrete particle swarm optimization algorithm, genetic algorithm and ant colony algorithm, DCSO algorithm is improved significantly in the solution accuracy, convergence speed and robustness. The feasibility of this algorithm is verified.

【Key words】 chicken swarm algorithm; 0-1 knapsack problem; discretization; adaptive mutation

0 引 言

随着国内外学者的不断研究，现阶段已涌现出了丰富多彩的群智能算法。其中较为经典的主要包括遗传算法（GA）、粒子群算法（PSO）、布谷鸟算法（CSO）等[1]，这些算法在一定程度上都促进了优化领域的发展。Kennedy 和 Eberhard 于 1997 年对基本粒子群算法离散化处理提出了二进制离散粒子群算法（BPSO）[2]；张晶等人采用二进制代码变换公式构建了二进制布谷鸟搜索算法（BCS）[3]；宋晓萍等人将杂草算法应用到离散问题，提出了一种离散杂草优化算法（DIWO）[4]；吴虎胜等人也提出了一种解决离散空间组合优化问题的二进制狼群算法（BWPA）[5]。上述算法具备各自的优劣性，因此，对于算法性能的研究仍然在不断地改进，与此同时，一些新颖的群智能算法也在不断地出现与发展。

鸡群算法（CSO）是由中国学者孟献兵于 2014 年在第五届群体智能国际会议中提出的一种群智能优化算法[6]。该算法模拟鸡群中的等级制度和觅食行为，具有很强的全局搜索能力。而如何将鸡群算法应用于离散空间，优化问题是鸡群算法的主要研究方向之一。本文设计出一种离散型鸡群算法（DCSO）并应用于 0-1 背包问题，在公鸡个体位置更新方式中，引入了自适应权重组合变异算子，通过仿真实验将该算法与遗传算法、粒子群算法、蚁群算法在解的质量、收敛速度等方面进行比较。

1 背包问题的数学模型

0-1 背包问题的一般描述为：假设有 n 个物品 $g_1, g_2, \cdots, g_n$ 和一个背包，第 j 件物品的重量及其价

基金项目：国家自然科学基金（11371015）；四川省教育厅自然科学基金（18ZA0469）；西华师范大学校级科研团队（CXTD2015-4）；西华师范大学英才基金（17YC385）。

作者简介：周 洋（1995-），女，硕士研究生，主要研究方向：数值计算、智能算法；潘大志（1974-），男，博士，教授，中国计算机学会（CCF）会员，主要研究方向：智能计算与算法设计。

收稿日期：2018-10-08

值分别为 $w_j>0$ 和 $v_j>0(j=1,2,\cdots,n)$，背包的最大载重限制为 C，选择部分物品放入背包，使得在不超过背包最大载重限制的前提下，所放入物品的总价值最大[7]。为描述 n 个物品是否放入背包，定义决策变量 x_j：

$$x_j=\begin{cases}1 & \text{物品 } g_j \text{ 放入背包}\\ 0 & \text{物品 } g_j \text{ 不放入背包}\end{cases}\quad j=1,2,\cdots,n \tag{1}$$

通过 $X=(x_1,x_2,\cdots,x_n)$ 表示物品的选择情况，此时放入物品的总重量是 $\sum_{j=1}^{n}w_jx_j$，总价值是 $\sum_{j=1}^{n}v_jx_j$，对应的数学模型为：

$$\max f(X)=\sum_{j=1}^{n}v_jx_j \tag{2}$$

$$\text{s.t.}\quad\begin{cases}\sum_{j=1}^{n}w_jx_j\leqslant C\\ x_j\in\{0,1\}\quad j=1,2,\cdots,n\end{cases} \tag{3}$$

2　基本鸡群算法

鸡群算法模拟鸡群中的等级制度和觅食行为。公鸡作为整个鸡群的首领主动去寻找食物；母鸡则陪伴在公鸡左右；小鸡们则跟随着鸡妈妈去寻找食物。根据这些行为，CSO 算法制定了以下规则[8]：

在一个鸡群中，包含有若干子种群，其中每个子种群中又包括一只公鸡、若干只母鸡和小鸡。

整个鸡群的划分以及鸡的身份角色确定都取决于适应度值。计算适应度并降序排列，选择适应度值最好的若干个体作为公鸡群体，适应度值最差的若干个体作为小鸡群体，剩余的若干个体就作为母鸡群体。其中，母鸡与公鸡之间的配偶关系、小鸡与母鸡之间的母子关系都是随机确定的。

在每一个子种群中，这些等级关系、配偶关系、母子关系将在一段时间内保持不变，每隔 G 代重新分群并更新角色。在每一个子种群中，鸡群跟随公鸡去寻找食物，并且保护自己的食物。小鸡则在母鸡妈妈身边寻找食物，适应度值越高的优势个体具有更强的竞争能力。

在 CSO 算法中，假设整个鸡群的数量为 N，公鸡的数量为 N_R，母鸡的数量为 N_H，称孵化小鸡的母鸡为妈妈母鸡，其数量为 N_M，小鸡的数量为 N_C。根据等级制度以及上述规则，可以确定公鸡、母鸡、小鸡的位置更新方式如下：

(1)公鸡的位置更新方式为：

$$x_{i,j}^{t+1}=x_{i,j}^{t}*(1+randn(0,\sigma^2)) \tag{4}$$

$$\sigma^2=\begin{cases}1 & \text{if } f_i<f_k\\ \exp\left(\dfrac{(f_k-f_i)}{|f_i|+\varepsilon}\right) & \text{其它}\end{cases}\quad k\in[1,N],k\neq i \tag{5}$$

其中，$x_{i,j}^{t}$ 为在第 t 次迭代时刻第 j 维空间的第 i 个个体；$randn(0,\sigma^2)$ 代表均值为 0，标准差为 σ 的高斯分布随机数；ε 是一个为了避免分母为 0 而取得很小的整数，一般取 $\varepsilon=1e-50$；f_i 代表第 i 个个体的适应度值，f_k 代表第 k 个个体的适应度值；k 代表从公鸡种群中随机选择的不同于公鸡 i 的公鸡。

(2)母鸡的位置更新方式为：

$$x_{i,j}^{t+1}=x_{i,j}^{t}+S_1*rand*(x_{r_1,j}^{t}-x_{i,j}^{t})+S_2*rand*(x_{r_2,j}^{t}-x_{i,j}^{t}) \tag{6}$$

$$S_1=\exp\left(\frac{f_{r_1}-f_i}{|f_i|+\varepsilon}\right) \tag{7}$$

$$S_2=\exp(f_{r_2}-f_i) \tag{8}$$

其中，$rand$ 表示产生[0,1]之间的随机数；r_1 表示第 i 只母鸡的配偶公鸡；r_2 表示从整个鸡群中随机选择的公鸡或者母鸡，并且 $r_1\neq r_2$。

(3)小鸡的位置更新方式为：

$$x_{i,j}^{t+1}=x_{i,j}^{t}+FL*(x_{m,j}^{t}-x_{i,j}^{t}) \tag{9}$$

其中，$x_{m,j}^{t}$ 表示在第 t 代时刻第 i 只小鸡的妈妈母鸡的位置；$FL\in[0,2]$，为小鸡跟随其妈妈母鸡的行走步长。

3　求解 0-1 背包问题的离散型鸡群算法

0-1 背包问题是运筹学中经典的组合优化问题之一，也是一类典型的 NP 难问题，下面对于离散化鸡群算法改进策略作简单阐述：

(1)对个体进行离散化处理[9]。

(2)针对基本鸡群算法中算法后期种群多样性降低这一缺陷，采用自适应权重组合变异更新公鸡的位置。

(3)采用定向变异策略增强小鸡对最优个体的学习，避免陷入局部最优。

(4)为了修复不可行解，每一代更新完成后都采用贪心修正算子(GMO)和贪心优化算子(GOO)进行修正与优化[10]。

3.1　离散化策略

为实现离散化，基于文献[11]中二进制粒子群算法中的离散化处理方式，对更新后的每个个体位置进行如下方式离散化处理：

$$x_{i,j}^{t}=\begin{cases}1 & \text{if } rand(\ ) > S(v_{ij})\\ 0 & \text{if } rand(\ ) < S(v_{ij})\end{cases} \tag{10}$$

其中，$S(x)=\dfrac{1}{1+\exp(-x)}$；$S(x)$ 为 *Sigmoid* 函数；$rand(\ )$ 为[0,1]区间内的随机数。

3.2 自适应权重组合变异

公鸡的位置更新方式实质上采用一种高斯变异算子，在整个迭代过程中，对于每个公鸡个体而言，变异能力一直不变。随着多样性的下降，很可能会使算法陷入局部最优解。为了加速算法收敛，考虑改进变异算子去更新公鸡位置，在高斯变异算子的基础上融入柯西变异算子[12]。图 1 直观地反映了高斯分布和柯西分布的概率密度函数曲线。

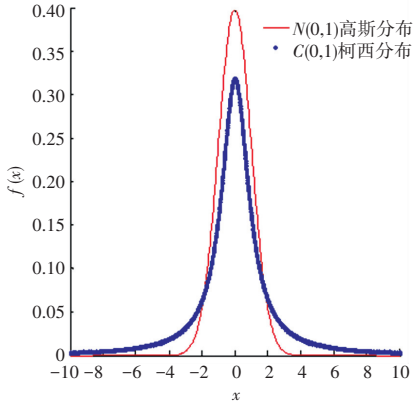


图 1 高斯分布和柯西分布的图形比较

Fig. 1 Graphical comparison between Guass distribution and Cauchy distribution

从图 1 可看出，在相同参数下，高斯分布在垂直方向上的中心峰值略高于柯西分布，但在水平方向上柯西分布两翼接近于 0 的速度却比高斯分布缓慢。也就是说，相比柯西变异算子而言，高斯变异算子对于当前点周围领域范围内的开采利用率更高，局部搜索能力更强；而柯西变异算子能够以更高的概率跳出局部最优解，全局搜索性能更高。

为了更好地平衡算法前期与后期的“开采”与“探索”，采用组合变异算子，同时引入进化代数 t 作为控制参数，动态调整变异算子中的变异步长 α，从而改进公鸡的位置更新方式为：

$$x_{i,j}^{t+1}=x_{i,j}^{t}*(1+\alpha*N(0,\sigma^2)+(1-\alpha)*C(0,1)) \tag{11}$$

$$\alpha=\eta*\exp\left(-\frac{t}{T}\right) \tag{12}$$

其中，$N(0,\sigma^2)$ 表示均值为 0，标准差为 σ 的高斯分布随机数；$C(0,1)$ 表示标准柯西分布随机数；α 代表权重系数；η 为小于 1 的正常数。多次试验得出 $\eta=0.5$ 时效果最好；T 为最大迭代次数。通过此方式对公鸡的变异进行动态扰动。

3.3 定向变异

基本鸡群算法中小鸡的位置更新仅仅吸收了其妈妈母鸡的位置信息，一旦其妈妈母鸡陷入局部最优解，那么小鸡也会陷入局部最优，不利于算法的收敛，采用定向变异策略改进小鸡位置更新方式为[13]：

$$x_{i,j}^{t+1}=best+FL*(x_{m,j}^{t}-x_{i,j}^{t}) \tag{13}$$

其中，*best* 为整个鸡群中的最优个体。这样小鸡在寻优时能够避免盲目搜索，提高了算法的稳定性。

3.4 修复机制

0-1 背包问题的实质为有约束优化问题，因此，在 DCSO 算法鸡群位置更新过程中，若新产生的鸡群个体不满足公式(3)，出现不可行解，将导致算法求解效率大大降低，则需要对其进行修正与优化。目前对于不可行解的处理方法主要有罚函数法和个体修正法。受文献[10]中贪心修正算子(GMO)和贪心优化算子(GOO)的启发，本文将这两种算子融入到 DCSO 中，不仅能对非正常编码个体进行修正，还能对所有个体进一步优化，快速找到最优解。

3.5 DCSO 算法的实现步骤

Step 1 随机初始化鸡群，设置相关参数 N，$N_R,N_H,N_M,N_C,G,\varepsilon$ 以及 0-1 背包数据的输入。

Step 2 当 $t=0$ 时，计算鸡群的适应度值 *fitness*，利用 GMO 和 GOO 算子修复不可行解并初始化当前个体最优 *pbest* 和全局最优 *gbest*。

适应度值表示方法为：$f(x_i)=(\sum_{j=1}^{D}p_j x_{i,j}^{t})$

Step 3 判断 $mod(t,G)=0$ 是否成立。若成立，则对个体适应度降序排列。根据第 3 节阐述的分群策略以及等级制度划分子群，并随机建立配偶关系和母子关系。

Step 4 利用式(11)、(6)和(13)分别对公鸡、母鸡和小鸡进行位置更新。

Step 5 当前迭代时刻中的鸡群全部更新完成后，采用 3.1 节中方式对所有个体的位置 $x_{i,j}$ 离散化处理。

Step 6 再次用 GMO 和 GOO 贪心算子对不可行解进行修正。

Step 7 重新计算所有个体的适应度值，并更新全局最优 *gbest*。

Step 8 令 $t=t+1$，当 $t>T$ 时，则停止迭代，输出最优解。否则返回 Step 3。

4　实验与仿真

为了验证 DCSO 算法的可行性与高效性，下面选用了被广泛使用的 4 个经典 0-1 背包问题来测试算法的寻优性能，其维数从 20 维至 100 维，可较全面地反映算法的性能。表 1 中已知最优解采用 A/B 的形式，A 为背包中物品总价值，B 为物品总重量。

表 1　0-1 背包问题的 5 个仿真实例

Tab. 1　5 simulation examples of 0-1 knapsack problem

编号	维数	参数(w,v,c)	已知最优解
$Q1$	20	w = [92,4,43,83,84,68,92,82,6,44,32,18,56,83,25,96,70,48,14,58], v = [44,46,90,72,91,40,75,35,8,54,78,40,77,15,61,17,75,29,75,63], c = 878	1 024/871
$Q2$	50	w = [438,754,699,587,789,912,819,347,511,287,541,784,676,198,572,914,988,4,355,569,144,272,531,556,741,489,321,84,194,483,205,607,399,747,118,651,806,9,607,121,370,999,494,743,967,718,397,589,193,369], v = [72,490,651,833,883,489,359,337,267,441,70,934,467,661,220,329,440,774,595,98,424,37,807,320,501,309,834,851,34,459,111,253,159,858,793,145,651,856,400,285,405,95,391,19,96,273,152,473,448,231], c = 11 258	16 102/11 231
$Q3$	80	w = [40,27,5,21,51,16,42,18,52,28,57,34,44,43,52,55,53,42,47,56,57,44,16,2,12,9,40,23,56,3,39,16,54,36,52,5,53,48,23,47,41,49,22,42,10,16,53,58,40,1,43,56,40,32,44,35,37,45,52,56,40,2,23,49,50,26,11,35,32,34,58,6,52,26,31,23,4,52,53,19], v = [199,194,193,191,189,178,174,169,164,164,161,158,157,154,152,152,149,142,131,125,124,124,124,122,119,116,114,113,111,110,109,100,97,94,91,82,82,81,80,80,80,79,77,76,74,72,71,70,69,68,65,65,61,56,55,54,53,47,47,46,41,36,34,32,32,30,29,29,26,25,23,22,20,11,10,9,5,4,3,1], c = 1 173	5 183/1 170
$Q4$	100	w = [54,95,36,18,4,71,83,16,27,84,88,45,94,64,14,80,4,23,75,36,90,20,77,32,58,6,14,86,84,59,71,21,30,22,96,49,81,48,37,28,6,84,19,55,88,38,51,52,79,55,70,53,64,99,61,86,1,64,32,60,42,45,34,22,49,37,33,1,78,43,85,24,96,32,99,57,23,8,10,74,59,89,95,40,46,65,6,89,84,83,6,19,45,59,26,13,8,26,5,9], v = [297,295,293,292,291,289,284,284,283,283,281,280,279,277,276,275,273,264,260,257,250,236,236,235,235,233,232,232,228,218,217,214,211,208,205,204,203,201,196,194,193,193,192,191,190,187,187,184,184,184,181,179,176,173,172,171,160,128,123,114,113,107,105,101,100,100,99,98,97,94,94,93,91,80,74,73,72,63,63,62,61,60,56,53,52,50,48,46,40,40,35,28,22,22,18,15,12,11,6,5], c = 3 820	15 170/3 818

为了保证测试的客观性与公平性，设置种群规模为 100，最大迭代次数为 200。仿真实验所选择的测试环境为：处理器为 Intel(R) Xeon(R) CPU E3-1240 v5,3.50 GHz，内存为 8G，操作系统为 Windows10，利用 Matlab2014a 编程求解。根据表 1 中的背包数据，对比以下 4 种算法，参数设置见表 2，每种算法各运行 30 次，从最优解、最差解、平均值、成功率等指标来分析，结果见表 3。

从表 3 可以看出，在求解 4 个经典 0-1 背包问题时，DCSO 算法都能找到理论最优解，特别是对于 $Q1$ - $Q4$，该算法都能在较低的迭代次数中求得问题最优解，且对于 $Q3$ 和 $Q4$ 而言，无论从标准差还是成功率来分析，DCSO 算法的求解精度和稳定性都远远优于 BPSO 算法。

表 2　4 种算法的参数设置

Tab. 2　Parameter setting of 4 algorithms

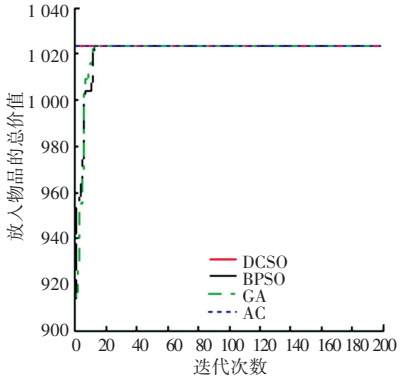
算法	参数
DCSO	$N_R = 0.2 * N$, $N_H = 0.6 * N$, $N_C = N - N_R - N_H$, $N_M = 0.1 * N_H$, $G = 10$
BPSO	$\omega = 0.8$, $c_1 = c_2 = 0.7$
GA	$P_c = 0.8$, $P_m = 0.01$
AC	$\alpha = 0.9$, $\beta = 1$, $\rho = 0.7$, $Q = 1$

为了进一步说明 4 种算法的寻优性能，图 2 给出了各算法迭代收敛曲线。从图中可以直观地看出，相比于其它 3 种算法，DCSO 均表现出了优越的收敛性能。综合所有指标充分说明本文提出的新算法具有较强的寻优能力，适合于求解 0-1 背包问题，具有可行性。

表 3 0-1 背包问题的 4 个仿真实例的结果对比

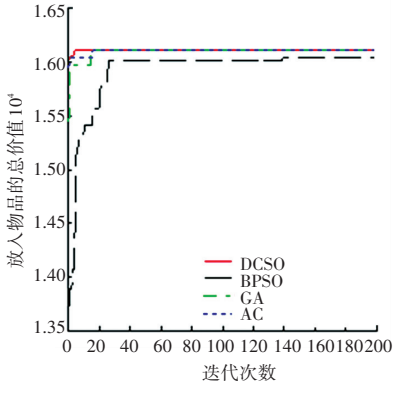
Tab. 3 Comparison of 4 simulation cases of 0-1 knapsack problem

编号	维数	算法	理论最优解	实验最优解	实验最差解	平均值	标准差	成功率/%	平均迭代次数
Q1	20	DCSO	1 024/871	1 024/871	1 024/871	1 024	0	100	2
		BPSO		1 024/871	1 024/871	1 024	0	100	5
		GA		1 024/871	1 024/871	1 024	0	100	12
		AC		1 024/871	1 024/871	1 024	0	100	6
Q2	50	DCSO	16 102/11 231	16 102/11 231	16 102/11 231	16 102	0	100	3
		BPSO		16 102/11 231	16 102/11 231	16 102	0	100	17
		GA		16 102/11 231	16 102/11 231	16 102	0	100	25
		AC		16 102/11 231	16 102/11 231	16 102	0	100	5
Q3	80	DCSO	5 183/1 170	5 183/1 170	5 181/1 169	5 181.93	0.99	46.67	81
		BPSO		5 183/1 170	5 097/1 123	5 138.93	22.97	6.67	162
		GA		5 079/1 170	4 907/1 145	5 013.53	56.08	0	119
		AC		5 042/1 056	4 308/982	4 857.47	164.57	0	189
Q4	100	DCSO	15 170/3 818	15 170/3 818	15 160/3 814	15 167.77	4.03	80	91
		BPSO		14 338/3 818	13 737/3 731	13 969.03	153.45	0	48
		GA		13 598/3 748	12 681/3 556	13 161.03	189.37	0	63
		AC		13 410/3 763	12 988/3 674	13 243.27	175.07	0	57



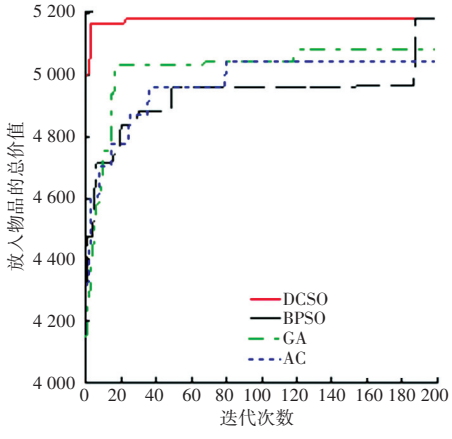
(a) $Q1$ 在 T = 200 时的收敛曲线比较

(a) Comparison of convergence curves of $Q1$ at T = 200



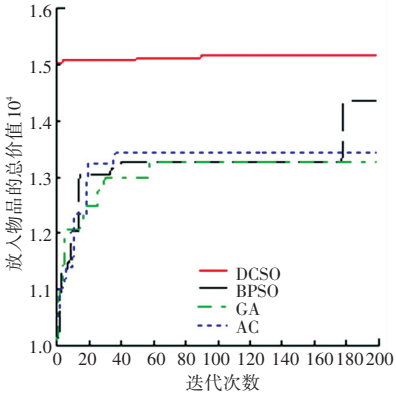
(b) $Q2$ 在 T = 200 时的收敛曲线比较

(b) Comparison of convergence curves of $Q2$ at T = 200



(c) $Q3$ 在 T = 200 时的收敛曲线比较

(c) Comparison of convergence curves of $Q3$ at T = 200



(d) $Q4$ 在 T = 200 时的收敛曲线比较

(d) Comparison of convergence curves of $Q4$ at T = 200

图 2 DCSO、BPSO、GA 和 AC 对 4 个仿真实例的收敛曲线比较

Fig. 2 Comparison of convergence curves of DCSO, BPSO, GA and AC for 4 simulation cases

5　结束语

在基本鸡群算法的基础上，基于离散思想，在公鸡位置更新中引入自适应权重变异算子，对小鸡的位置更新采用定向变异的策略，与此同时，融入贪心修复算子对不可行解进行修复与优化，提出了一种求解 0-1 背包问题的离散型鸡群算法。通过 4 个经典 0-1 背包实例的仿真实验，对比 BPSO、GA、AC 等算法，显著提高了解的质量，验证了该算法的可行性以及高效性。

参考文献

[1] RAJABIOUN R.Cuckoo optimization algorithm[J]. Applied Soft Computing,2011,11(8):5508-5518.

[2] KENNEDY J, EBERHART R C. A discrete binary version of the particle swarm algorithm [C]//1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation. Orlando, FL, USA: IEEE ,1997:4104-4108.

[3] 张晶,吴虎胜. 改进二进制布谷鸟搜索算法求解多维背包问题[J]. 计算机应用,2015,35(1):183-188.

[4] 宋晓萍,胡常安. 离散杂草优化算法在 0/1 背包问题中的应用[J]. 计算机工程与应用,2012,48(30):239-242,248.

[5] 吴虎胜,张凤鸣,战仁军,等. 求解 0-1 背包问题的二进制狼群算法[J]. 系统工程与电子技术,2014,36(8):1660-1667.

[6] MENG Xianbing, LIU Yu, GAO Xiaozhi, et al. A new bio-inspired algorithm: Chicken swarm optimization[M]// TAN Y, SHI Y, COELLO C A C. Advances in Swarm Intelligence. ICSI 2014. Lecture Notes in Computer Science. Cham:Springer,2014,8794:86-94.

[7] 田烽楠,王于. 求解 0-1 背包问题算法综述[J]. 软件导刊,2009,8(1):59-61.

[8] 杨小健,徐小婷,李荣雨. 求解高维优化问题的遗传鸡群优化算法[J]. 计算机工程与应用,2018,54(11):133-139.

[9] FOURIE P C, GROENWOLD A A. The particle swarm optimization algorithm in size and shape optimization[J]. Structural and Multidisciplinary Optimization, 2002, 23(4):259-267.

[10] 贺毅朝,宋建民,张敬敏,等. 利用遗传算法求解静态与动态背包问题的研究[J]. 计算机应用研究,2015,32(4):1011-1015.

[11] 马慧民,叶春明,张爽. 二进制改进粒子群算法在背包问题中的应用[J]. 上海理工大学学报,2006,28(1):31-34.

[12] 周方俊,王向军,张民. 基于 t 分布变异的进化规划[J]. 电子学报,2008,36(4):667-671.

[13] 刘建芹,贺毅朝,顾茜茜. 基于离散微粒群算法求解背包问题研究[J]. 计算机工程与设计,2007,28(13):3189-3191,3204.

(上接第 97 页)

[3] 潘涛, 张弛, 杜国明, 等. 城乡不透水面增长格局及地表温度的响应特征研究[J]. 地球信息科学学报, 2017, 19(1):134-142.

[4] 张旸, 胡德勇, 陈姗姗. 北京城区不透水地表盖度变化及对地表温度的影响[J]. 地球信息科学学报, 2017, 19(11):1504-1513.

[5] 郑飞, 张殿发, 孙伟伟, 等. 基于 ASTER 遥感的杭州城市热/冷岛的景观特征分析[J]. 遥感技术与应用, 2017, 32(5):938-947.

[6] 刘焱序, 彭建, 王仰麟. 城市热岛效应与景观格局的关联:从城市规模、景观组分到空间构型[J]. 生态学报, 2017, 37(23):7769-7780.

[7] 王刚, 管东生. 植被覆盖度和归一化湿度指数对热力景观格局的影响—以广州为例[J]. 应用生态学报, 2012, 23(9):2429-2436.

[8] 李苗苗, 吴炳方, 颜长珍, 等. 密云水库上游植被覆盖度的遥感估算[J]. 资源科学, 2004, 26(4):153-159.

[9] 王娜. 城市热环境和植被覆盖关系的方法研究[D]. 上海:上海师范大学, 2009.

[10] 吴彦樨, 焦利民, 金健飞. 武汉市城市热岛与植被覆盖的空间相关性分析[J]. 环境科学与技术, 2016,39(11):156-161.

[11] 张晓娟, 周启刚, 黄丽昕, 等. 重庆市主城区地表温度与植被覆盖指数关系研究[J]. 土壤通报, 2018,49(2):293-302.

[12] 秦立厚, 张茂震, 袁振花, 等. 基于人工神经网络与空间仿真模拟的区域森林碳估算比较—以龙泉市为例[J]. 生态学报, 2017, 37(10):3459-3470.

[13] 覃志豪, ZHANG Minghua, KARNIELI A. 用陆地卫星 TM6 数据演算地表温度的单窗算法[J]. 地理学报, 2001, 56(4):456-466.

[14] ROZENSTEIN O, QIN Z, DERIMIAN Y, et al. Derivation of land surface temperature for Landsat-8 TIRS using a split window algorithm[J]. Sensors, 2014, 14(4):5768-5780.

[15] 谢元礼, 范熙伟, 韩涛, 等. 基于 TM 影像的兰州市地表温度反演及城市热岛效应分析[J]. 干旱区资源与环境, 2011, 25(9):172-175.

[16] 胡德勇, 乔琨, 王兴玲, 等. 利用单窗算法反演 Landsat 8 TIRS 数据地表温度[J]. 武汉大学学报(信息科学版), 2017, 42(7): 869-876.

[17] 宋挺, 段峥, 刘军志, 等. Landsat 8 数据地表温度反演算法对比[J]. 遥感学报, 2015, 19(3):451-464.

[18] 姚小薇, 曾杰, 李旺君. 武汉城市圈城镇化与土地生态系统服务价值空间相关特征[J]. 农业工程学报, 2015, 31(9):249-256.

[19] JAGANMOHAN M, KNAPP S, BUCHMANN C M, et al. The bigger, the better? The influence of urban green space design on cooling effects for residential areas[J]. Journal of Environmental Quality, 2016, 45(1):134-145.

[20] 王刚, 张秋平, 肖荣波, 等. 城市绿地对热岛效应的调控功能研究—以广州为例[J]. 生态科学, 2017, 36(1):170-176.

[21] 王琳, 祝亚鹏, 卫宝立, 等. 快速发展的中小城市地表热环境及水体温度调控作用研究—以山东省滨州市为例[J]. 水土保持通报, 2018,38(2):102-109.