文章编号: 1000-5641(2020)06-0090-09

求解 0-1 背包问题的混合粒子群改进算法研究

姚若侠, 薛 丹, 谢娟英, 范 虹

(陕西师范大学 计算机科学学院, 西安 710119)

摘要: 针对 0-1 背包问题求解, 将离散二进制粒子群优化 (Binary Particle Swarm Optimization, BPSO) 算法、贪心优化策略和模拟退火算法有机结合, 提出了一种改进算法: 带贪心优化的混合粒子群和模拟退火 (Hybrid optimization algorithm based on the BPSO, the Simulated Annealing (SA) Algorithm and the Combined Greedy Optimization Operator (CGOO), BPSOSA-CGOO) 算法. 基于新算法, 完成了 9 组不同维度数据的仿真实验. 实验结果表明, BPSOSA-CGOO 算法能够以较小的种群规模及迭代次数实现 0-1 背包问题的有效求解, 并在问题维度为 20 维的测试数据中找到优于已知最优解的解; 独立重复实验验证了, 无论对于低维度还是高维度背包问题, BPSOSA-CGOO 算法均能以较高概率命中最优解, 提高了高维度背包问题求解的稳定性和可靠性.

关键词: 背包问题; 粒子群优化算法; 贪心优化策略; 模拟退火算法

中图分类号: TP301 **文献标志码**: A **DOI**: 10.3969/j.issn.1000-5641.201921025

Study of an improved hybrid particle swarm optimization algorithm for solving 0-1 knapsack problems

YAO Ruoxia, XUE Dan, XIE Juanying, FAN Hong

(*School of Computer Science, Shaanxi Normal University, Xi'an* 710119, *China*)

Abstract: In order to solve 0-1 knapsack problems with greater stability and efficiency, we propose a BPSOSA-CGOO (Hybrid optimization algorithm based on the BPSO (binary particle swarm optimization), the simulated annealing (SA) algorithm and the combined greedy optimization operator (CGOO)) algorithm which is composed of the BPSO, the greedy optimization strategy, and the simulated annealing algorithm. Simulation experiments of 9 groups of different dimensions show that the BPSOSA-CGOO algorithm can solve 0-1 knapsack problems efficiently for small population sizes and iteration times. Meanwhile, it was observed that experiments performed with the algorithm can also find a better solution for 20-dimensional test data. In independent and repeated experiments, the BPSOSA-CGOO algorithm can achieve the optimal solution with high probability for both low-dimensional and high-dimensional knapsack problems; hence, stability and reliability is significantly improved when using the BPSOSA-CGOO algorithm to solve high-dimensional knapsack problems.

Keywords: knapsack problem; particle swarm optimization algorithm; greedy optimization strategy; simulated annealing algorithm

收稿日期: 2019-12-06

基金项目: 国家自然科学基金 (11471004, 61673251); 陕西省重点研究和发展项目 (2018SF-251)

通信作者: 姚若侠, 女, 教授, 博士生导师, 研究方向为符号计算、智能计算、模式识别. E-mail: rxyao@snnu.edu.cn

0　引　　言

0-1 背包问题 (0-1 Knapsack Problem, 0-1 KP) 是经典的组合优化问题, 也是一个 NP(Non-deterministic Polynomial)-难问题[1]. 背包问题具有深远的理论意义和重要的实践应用研究价值, 自 20 世纪 50 年代被 Dantzig 提出以来, 引起了国内外学者极大的研究热情, 且已被广泛应用于投资决策、资源分配和货物装载等金融和工业领域[2-6].

0-1 KP 简要描述如下: 共有 N 种待选定的物品, 每种物品只有 1 个且不可分割, 物品 j 的价值为 p_j, 重量为 h_j, 只有 1 个背包且能容纳物品的总重量为 C. 问题: 如何选择装入这个背包的物品, 使背包中装入物品的总重量不超过 C 且总价值最大? 0-1 KP 的数学模型公式[7] 为

$$\max f(x)=\sum_{j=1}^{n}p_j\times x_j,\ \ \sum_{j=1}^{n}h_j\times x_j\leqslant C,\ \ x_j\in\{0,1\},\ \ j=1,2,\cdots,n, \tag{1}$$

其中, $f(x)$ 表示装入背包物品的总价值, 当 $x_j=0$ 时, 表示物品 j 不装入背包, 当 $x_j=1$ 时, 表示物品装入背包.

目前求解 0-1 KP 的主要算法大致分为精确算法、群智能优化算法、混合算法[8-10] 这 3 类. 精确算法主要包括贪心算法[11](Greedy Algorithm, GA)、动态规划 (Dynamic Programming, DP) 算法[11]、分支界定 (Branch and Bound, BB) 算法[11]. 群智能优化算法主要包括遗传算法[12-15](Genetic Algorithm, GA)、烟花算法[16](Fireworks Algorithm, FA)、粒子群优化算法[17-20](Particle Swarm Optimization Algorithm, PSO) 等. 混合算法主要包括贪心遗传算法[21-22](Greedy Genetic Algorithm, GGA)、贪心萤火虫算法[23]、基于直觉模糊熵的粒子群-模拟退火[24](Intuitionistic Fuzzy Entropy Particle Swarm Optimization-Simulated Annealing, IFEPSO-SA) 算法、贪心优化粒子群算法 (Greedy Optimization Particle Swarm Optimization Algorithm, GOPSO)[25] 等. 文献 [11] 应用贪心算法、动态规划算法、分支界定算法求解 0-1 KP, 其实验结果表明, 贪心算法高效但却不一定能给出最优解; 在问题维数较低的情况下, 动态规划、分支界定的算法性能较好, 但随着问题规模的增大, 时间复杂度呈指数级增长, 同时动态规划算法的内存消耗也很大[11], 这意味着它们对大规模 0-1 KP 的解决只是理论层面的. 为了以较低的时间和空间消耗来较优地解决高维度背包问题, 大量学者将目光转移向了群智能优化算法及其混合算法[8,21-25]. 文献 [8] 应用遗传算法等 6 个算法, 求解大量的 0-1 KP 测试数据, 实验结果表明, 混合遗传算法和模拟退火算法性能较优[8]. 文献 [21-22] 提出了贪心遗传算法 (GGA)[21], 并对文献 [11] 中的贪心策略进行了改进, 提出了贪心修正算子 (CMO)[22] 和贪心优化算子 (COO)[22], 使得应用遗传算法在求解 0-1 KP 时不仅可以对不合法的解进行校正, 也可以对合法解给出优化, 有效提升了种群在进化过程中优质解所占的比率, 增强了遗传算法求解 0-1 KP 的能力, 拓展了应用群智能优化算法求解组合优化问题的思路. 文献 [24] 结合粒子群优化算法和模拟退火策略, 提出了基于直觉模糊熵的粒子群-模拟退火算法[24], 通过实验验证了算法的有效性, 而且, 该算法对于较小规模的背包问题能够以 100% 的概率命中已知解. 文献 [25] 结合粒子群优化算法、贪心修正算子 (Greedy Modification Operator, GMO)[22] 和贪心优化算子 (Greedy Optimization operator, GOO)[22], 提出了贪心优化粒子群算法[25], 通过实验验证了该算法具有良好的寻优能力, 并且可以快速收敛至最优解. 以上研究成果显示, 混合群智能优化算法对求解 0-1 KP 具有良好的表现, 由此启发本文尝试采用带贪心优化的混合粒子群和模拟退火 (BPSOSA-CGOO) 算法来求解 0-1 KP.

基于对 0-1 KP 现状的研究和分析, 本文结合离散二进制粒子群优化算法 (BPSO)[26] 可以快速收敛至潜在最优解和对于组合优化问题有广泛适应性的特点、贪心修正算子 (GMO)[22] 和贪心优化算子 (GOO)[22] 能够在种群进化过程中提升优质解占有率的特点、模拟退火 (Simulated Annealing, SA)[27-29]

算法能够概率性地跳出局部极值等特点, 提出了带贪心优化的混合粒子群和模拟退火算法 (BPSOSA-CGOO), 拓展了求解 0-1 KP 更为稳定和有效的算法集. 通过对文献 [24] 中的背包数据进行测试, BPSOSA-CGOO 算法使求解 0-1 KP 的寻优能力和命中已知最优解的概率得到了有效的提高. 下文对 BPSOSA-CGOO 算法进行详细介绍.

1 带贪心优化的混合粒子群和模拟退火算法

本文提出的 BPSOSA-CGOO 算法, 主要在离散二进制粒子群优化算法 (BPSO)[26] 的基础上进行了以下两个优化.

(1) 在种群初始化及更新粒子后, 本文调用合并的贪心优化算子 (Combined Greedy Optimization Operator, CGOO) 对更新后的粒子进行优化, 以此保证在种群合法性的基础上价值最大, 使得种群在进化过程中优质解的占比增高, 进而提升种群的寻优能力.

(2) 在粒子根据适应度更新完个体极值后, 随机选择 1 个粒子的个体极值 $p_{\text{Best},i}^{(t)}$ 进行模拟退火操作, 其目的是利用模拟退火操作可以概率性地接收新解的特点, 即模拟退火操作在降温过程中, 根据马尔科夫链长, 可概率性地接收不优于当前解的新解, 探寻更多的未知解空间, 很好地克服了粒子群优化算法容易陷入局部极值的缺点.

1.1 离散二进制粒子群优化算法

BPSO 由 James Kennedy 和 Russell C. Eberhart 于 1997 年首次提出, 随后被广泛应用于离散空间的二进制优化搜索. BPSO 中, 每个粒子由 1 个二进制编码串组成, 如 $x_i=(x_{i1}x_{i2}\cdots x_{ij}\cdots x_{in})$, 每个粒子对应的速度为 $v_i=(v_{i1}v_{i2}\cdots v_{ij}\cdots v_{in})$, 粒子根据公式 (2) 更新速度[26], 其中, w 表示惯性权重, c_1、c_2 为学习因子, r_1、r_2 表示 (0, 1) 之间的随机数, $p_{\text{Best},ij}$ 表示个体极值, 代表某个粒子在进化中寻找到的最优解, $g_{\text{Best},j}$ 表示全局极值, 代表种群在进化过程中寻找到的最优解, 个体极值和全局极值由适应度函数确定, 在 0-1 KP 中, 适应度函数计算当前解对应的背包价值, 价值越高表示适应度越优. 粒子根据公式 (2) 更新速度后, BPSO 通过 sigmoid 函数将更新后的新速度映射到 [0, 1] 区间, 表示相应的 x_{ij} 取 1 的概率, 如公式 (3), 然后根据公式 (4) 更新粒子[26]. 具体公式如下.

$$v_{ij}^{t+1}=w\times v_{ij}^{t}+c_1r_1\left(p_{\text{Best},ij}^{t}-x_{ij}^{t}\right)+c_2r_2\left(g_{\text{Best},j}^{t}-x_{ij}^{t}\right), \tag{2}$$

$$\text{sig}\left(v_{ij}\right)=\frac{1}{1+\exp\left(-v_{ij}\right)}, \tag{3}$$

$$x_{ij}=\begin{cases}1, & \text{如果sig}\left(v_{ij}\right)\geqslant\text{random}(),\\ 0, & \text{其他.}\end{cases} \tag{4}$$

公式 (2) 中, 上标 t 代表第 t 次进化, x_{ij}^{t} 代表粒子 i 的第 j 个分量在第 t 次进化的取值.

1.2 合并的贪心优化算子

合并的贪心优化算子 CGOO 是对 GMO 算子[22] 和 GOO 算子[22] 的合并, 主要实现对种群中不合法解的纠正, 以及对所有解的优化. 将商品的价值与重量的比值记为 $v\left(p_i\right)=\frac{p_i}{h_i}$, 其中, p_i、h_i 分别表示第 i 个商品的价值和重量. 具体执行过程如下.

(1) 按照 $v\left(p_i\right)$ 由大到小的顺序对粒子已选择的商品进行检查, 如果已选择的商品加入背包后, 不大于背包总承重, 则加入背包, 否则, 取出该商品.

(2) 按照 $v\left(p_i\right)$ 由大到小的顺序对粒子未选择的商品进行尝试性地加入背包, 如果未选择的商品加入背包后, 不大于背包总承重, 则加入背包, 否则, 不加入.

CGOO 算子优化算法见算法 1.

算法 1 CGOO 算子优化算法

输入:粒子 $X=(x_1\cdots x_j\cdots x_n)$;

重量集 $H=\{h_1, \cdots, h_j, \cdots, h_n\}$;

商品价值/重量的比值 $v(p_i)$,按由大到小排序的数组 $S=\{s_0, s_2, \cdots, s_{n-1}\}$;

背包总重量 C

输出:修正后的粒子 $X=(x_1\cdots x_j\cdots x_n)$

```
1: c_w = 0 //当前背包重量
2: //按照 v(p_i) 由大到小的顺序, 对已加入背包的商品进行检查, 如果加入后满足背包容量, 则加入, 否则取出
3: for k = 0 to n−1 do
4:     if X_S[k] == 1 then
5:         if c_w + H_S[k] <= C then
6:             c_w += H_S[k]
7:         else
8:             X_S[k] = 0
9:         end if
10:    end if
11: end for
12: //对没有加入背包的物品, 按照 v(p_i) 由大到小开始尝试性加入, 如果满足背包重量, 则加入
13: for k=0 to n−1 do
14:    if X_S[k] == 0 then
15:        if c_w + H_S[k] <= C then
16:            X_S[k] = 1
17:            c_w+= H_S[k]
18:        end if
19:    end if
20: end for
```

为了更清楚地描述 BPSOSA-CGOO 算法中 CGOO 算子的优化过程, 本文以 1 个 10 维背包问题为例, 列举某不合法粒子 X 经过 CGOO 算子优化后的结果, 结果如下.

设 10 个物品的价值集 $P=\{55, 10, 47, 5, 4, 50, 8, 61, 85, 87\}$, 重量集 $H=\{95, 4, 60, 32, 23, 72, 80, 62, 65, 46\}$, 背包最大容量 $C=269$, 粒子 $X=(1011100010)$, 粒子 X 优化前, 重量为 $275(>C)$, 价值为 196. 算法 1 执行后, 粒子 $X=(1110100010)$, 重量为 $247(<C)$, 价值为 $201(>196)$.

1.3　模拟退火策略

模拟退火操作的基本思想由 Metropolis 于 1953 年提出, 1983 年首次应用于解决组合优化问题[27]. 目前, 模拟退火操作有许多改进版本[28-29], 本文选用最基本的模拟退火策略, 即在降温的过程中根据马尔科夫链长, 重复执行如下迭代过程: ① 产生新解; ② 计算价值差; ③ 判断是否接收新解; ④ 接收或舍弃; ⑤ 返回①.

基于对算法复杂度的考虑, BPSOSA-CGOO 算法采用从种群中随机选择 1 个粒子的个体极值 $p_{\mathrm{Best},i}^{(t)}$ 进行模拟退火操作的策略, 其中产生新解的策略如下.

(1) 产生 1 个 (1, $N/3$) 之间的随机整数 flipNum 代表 $p_{\mathrm{Best},i}^{(t)}$ 中要更改的位数.

(2) 在 $p_{\mathrm{Best},i}^{(t)}$ 中随机选择 flipNum 位进行逻辑“非”操作, 从而产生新解.

在降温过程中根据马尔科夫链概率性地接收新解, 直至温度达到冷冻点. 图 1 是模拟退火操作产生新解的示意图.

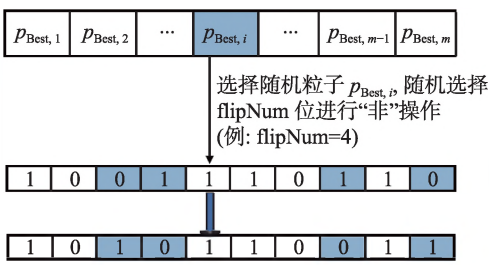

图 1 模拟退火操作产生新解示意图

Fig. 1 Schematic diagram of a simulated annealing operation generating new solutions

BPSOSA-CGOO 算法不仅可以保证粒子在进化过程中都合法, 提高种群中优质解的占有率, 而且还可以通过模拟退火操作概率性地修改个体极值, 有助于种群在进化过程中趋于全局最优. BPSOSA-CGOO 算法的程序流程如图 2 所示.

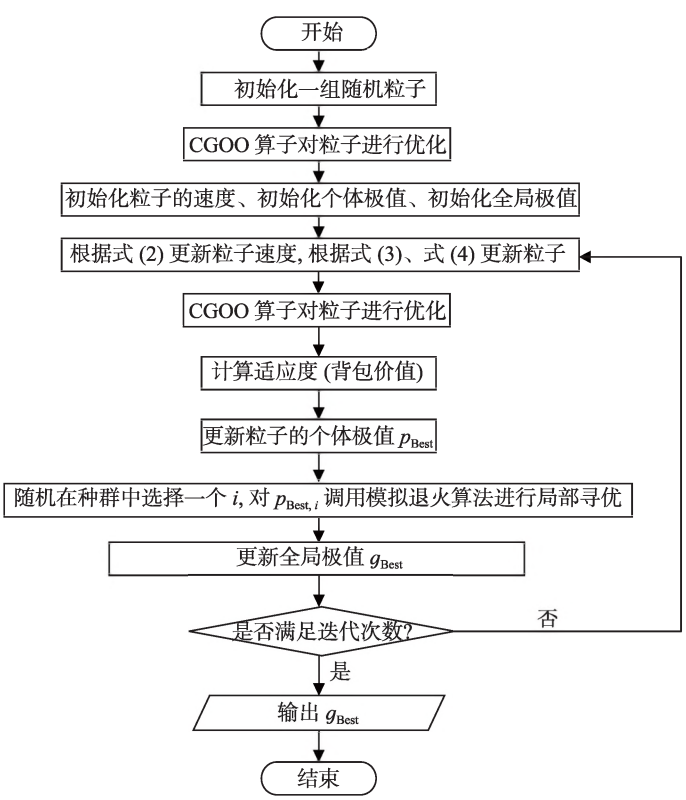

图 2 BPSOSA-CGOO 算法流程图

Fig. 2 Flow chart of the BPSOSA-CGOO algorithm

2 仿真实验

实验环境为, Win 10 操作系统, Intel®Core(TM)i7-8550U CPU 处理器, 8 GB 内存, Python3.5 编程语言, PyCharm2018.1.4 开发环境. 实验参数设置为, 种群规模=问题维数/2, 迭代次数=200, 学习因子 $c_1 = c_2 = 1$, 惯性系数 $w = 0.9$, 速度的最大值 $V_{\max} = 6$, 初始温度 $T_0 = 1\ 000$, 马尔科夫链长

$L=20$, 降温因子 $\alpha=0.99$, 冷冻点 $f=0.000\ 1$. 采用文献 [24] 中的 9 组不同维度的测试数据, 对 9 组测试数据各执行 1 次, 实验结果见表 1, 其中进化次数代表收敛至已知最优解的进化次数.

表 1　BPSOSA-CGOO 算法对不同测试数据的执行结果

Tab. 1　Results from the BPSOSA-CGOO algorithm with different test data

编号	维数	已知解		实验最优解	实验解		进化次数
		价值	重量		价值	重量	
1	10	295	269	(0111000111)	295	269	1
2	15	481.07	354.96	(001010110111011)	481.07	354.96	1
3	**20**	**1024**	**871**	**(10111111010111111101)**	**1042**	**878**	**3**
4	23	9767	9768	(11111111010000011000000)	9767	9768	12
5	50	3103	1000	(11010101111011011011011111110100001010011000001000)	3103	1000	63
6	50	16102	11231	(01111001010111000110101010110100011011011010000110)	16102	11231	14
7	60	8362	2393	(111111111111111111111111111111111111111111110100110110111000)	8362	2393	10
8	80	5183	1170	(11111111111111111111111111101110101001000101100010000000000010000100000000000)	5183	1170	103
9	100	15170	3818	(11111111111111111111111111111111111111111111111111111111111111111111111111111111010101111000011010001100011010)	15170	3818	10

表 1 实验结果揭示, BPSOSA-CGOO 算法对于 9 组不同维度的测试用例, 均能搜索到已知最优解, 且测试数据 3(表 1 加粗部分) 搜索到的最优解 (1042(价值) 878(重量)) 优于文献 [24] 中的解 (1024(价值) 871(重量)).

为了进一步分析算法的稳定性和有效性, 本文在实验参数设置不变的基础上, 对相同的测试数据, 分别执行了 20 次独立重复实验, 实验结果见表 2. 表 2 中, S_{SN} 代表成功实验的次数, A_{VIN} 代表成功实验中平均进化代数. 假定认为命中已知最优解为一次成功实验, 则 A_{VIN} 的计算公式为

$$A_{\mathrm{VIN}}=\sum_{i=1}^{S_{\mathrm{SN}}}\frac{E_i}{S_{\mathrm{SN}}}, \tag{5}$$

其中, E_i 为第 i 次成功实验对应的进化代数.

表 2 所示的实验结果揭示了, 在 9 组不同测试数据的 20 次独立重复实验结果中, 有 7 组实验命中最优解的次数都为 20 次, 标准差为 0; 测试数据 5 命中最优解的次数为 18 次, 标准差为 2.48; 测试数据 8 命中最优解 1 次, 标准差为 13.82. 这说明: ① BPSOSA-CGOO 算法对于大多数测试数据能以较高的概率命中最优解, 体现了有效性; ② 标准差是反映数据集离散程度的统计量, 表 2 中的标准差数据表明, BPSOSA-CGOO 算法具有较强的稳定性.

为了更加全面地分析 BPSOSA-CGOO 算法求解 0-1 背包问题的稳定性, 本文基于文献 [25] 中的算法思想对 GOPSO[25] 算法进行了 Python 实现, 并对上述 9 组不同维度的测试用例分别执行了 20 次独立重复实验. 实验环境为本文上述环境; 实验参数设置为, 种群规模=问题维数/2, 迭代次数=200, 学习因子 $c_1=c_2=1$, 惯性系数 $w=0.9$, 速度最大值 $V_{\max}=6$. 本文所提的 BPSOSA-CGOO 算法命中最优解的次数与 IFEPSOSA[24] 算法和 GOPSO[25] 算法的对比见图 3.

图 3 表明, GOPSO 算法、BPSOSA-CGOO 算法除测试数据 8 以外, 均能以较高的次数命中最优解. 针对 GOPSO 和 BPSOSA-CGOO 算法在测试数据 8 上表现出的特殊情况, 本文做了分析并给出先验推测: 这很可能是由种群规模引起的. 故本文尝试将种群规模=问题维数/2 修改为与文献 [24] 一致, 即种群规模 = 100, 其余参数保持不变, 对测试数据 8 执行 20 次独立重复实验, 执行结果见表 3 和图 4, 命中最优解次数与文献 [24] 中 IEFPSOSA[24] 算法的对比见图 5.

表 2 BPSOSA-CGOO 算法 20 次独立重复实验执行结果

Tab. 2 Results from the BPSOSA-CGOO algorithm with different test data after 20 repeat runs

编号	维数	已知解		最优解		最差解		平均值		标准差	S_{SN}	A_{VIN}
		价值	重量	价值	重量	价值	重量	价值	重量			
1	10	295.00	269.00	295.00	269.00	295.00	269.00	295.00	269.00	0.00	20	0.8
2	15	481.07	354.96	481.07	354.96	481.07	354.96	481.07	354.96	0.00	20	1.7
3	20	1024.00	871.00	1042.00	878.00	1042.00	878.00	1042.00	878.00	0.00	20	14.7
4	23	9767.00	9768.00	9767.00	9768.00	9767.00	9768.00	9767.00	9768.00	0.00	20	12.0
5	50	3103.00	1000.00	3103.00	1000.00	3093.00	1000.00	3102.20	1000.00	2.48	18	100.0
6	50	16102.00	11231.00	16102.00	11231.00	16102.00	11231.00	16102.00	11231.00	0.00	20	51.0
7	60	8362.00	2393.00	8362.00	2393.00	8362.00	2393.00	8362.00	2393.00	0.00	20	11.3
8	80	5183.00	1170.00	5183.00	1170.00	5135.00	1172.00	5170.50	1169.90	13.82	1	103.0
9	100	15170.00	3818.00	15170.00	3818.00	15170.00	3818.00	15170.00	3818.00	0.00	20	14.4

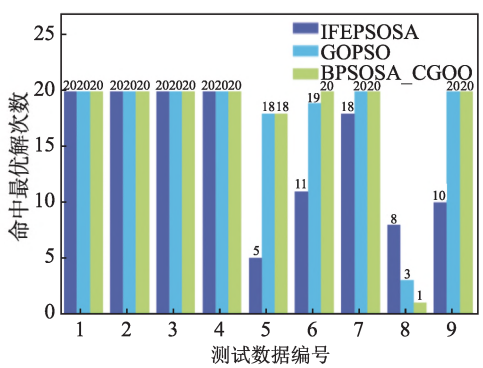

图 3 不同测试数据命中次数对比柱状图

Fig. 3 Comparison bar graph of the hit times with different test data

表 3 修改参数后测试数据 8 的 20 次独立重复实验执行结果

Tab. 3 Results from 20 independent experiments of test data 8 after parameter modification

算法	已知解	最优解	最差解	平均值	标准差	S_{SN}
GOPSO	5183	5183	5168	5180.7	3.86	8
BPSOSA-CGOO	5183	5183	5178	5182.2	1.32	13

表 3 所示的实验结果揭示, BPSOSA-CGOO 算法的最差解、平均值、标准差均优于 GOPSO 算法. 图 4 表明, BPSOSA-CGOO 算法在 20 次独立重复实验中找到的实验最优解的数值波动较小, 较为稳定. 由图 5 可以看出, BPSOSA-CGOO 算法在独立重复实验中命中已知最优解 13 次, 体现了该算法在命中最优解的次数上的优势.

此外, 为了充分验证 BPSOSA-CGOO 算法在高维度背包问题求解上的稳定性, 本文对文献 [16] 中 100 维的背包算例进行了 10 次独立重复实验, 实验结果见表 4. 表 4 所示的实验结果揭示, 虽然 BPSOSA-CGOO 算法的实验最优解稍小于文献 [16] 中的已知解, 但独立重复实验的标准差远远小于文献 [16] 中的 147, 实验最差解大于文献 [16] 中的 7 753, 且随着种群规模和迭代次数的增大, BPSOSA-CGOO 算法标准差随之减小, 最差解也在增大. 这表明, 对于高维度背包问题, BPSOSA-CGOO 算法能够以较高的稳定性找出较优解.

上述实验数据及结果表明, BPSOSA-CGOO 算法不仅可以有效求解 0-1 背包问题, 而且能够在种

群规摸、迭代次数均较小的条件下, 对高维度背包问题以较高概率寻得最优解/较优解, 表现出了较强的稳定性和可靠性.

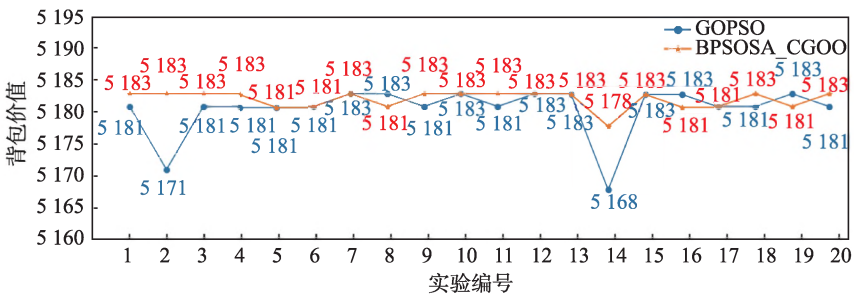

图 4　20 次独立重复执行测试数据 8 的结果图

Fig. 4　Result chart of test data 8 in 20 independent experiments

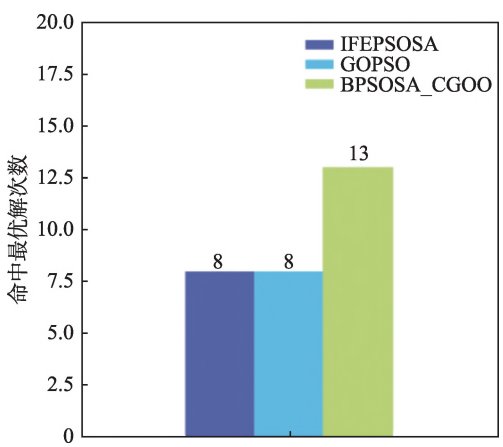

图 5　测试数据 8 命中次数对比柱状图

Fig. 5　Comparison bar graph of hit times in test data 8

表 4　10 次独立重复实验执行结果

Tab. 4　Results from 10 independent experiments

用例维数	实验参数		已知解	最优解	最差解	平均值	标准差
	种群	迭代					
100	100	200	8254	8003	7925	7957.5	21
	1000	500	8254	8016	7997	8010.7	6

3　结束语

本文基于对离散二进制粒子群优化算法的两个优化, 提出了带贪心优化的混合粒子群和模拟退火算法——BPSOSA-CGOO 算法, 采用该算法求解 0-1 背包问题, 完成了 9 组不同维度背包数据的测试后, 得到了以下结论: ① BPSOSA-CGOO 算法对于 9 组背包数据均能以较小的种群和进化代数找到已知最优解, 且背包数据 3 找到的解优于已知最优解; ② 在独立重复实验中, 有 7 组实验以 100% 的概率命中最优解, 其余两组命中最优解的概率分别为 90% 和 65%.

BPSOSA-CGOO 算法在本文实验环境下, 无论对于高维度还是低维度的测试数据, 均能以较高的概率命中最优解, 在求解 0-1 KP 的稳定性、可靠性方面, 性能均获得明显提升. 此外, 谷鹏等[30] 根据粒子群优化算法提出了一种改进的粒子群优化算法, 并结合扩展的卡尔曼滤波跟踪算法, 提高了目标

跟踪的精度,该算法虽然能够随机搜索潜在最优解,但是容易陷入局部最优的特点. 李鹏等[31]通过将粒子群优化算法与遗传算法相结合,在种群中进行交叉、变异等操作提高了种群多样性,克服了粒子群优化算法容易陷入局部极值的缺点,并将其成功应用于无人机航路规划中,取得了良好的应用效果. 郭玉洁等[32]提出了一种双种群协同多目标粒子群优化算法,将其应用于油田开采,以实现油田开采利润最大化为目标,并应用于油田开采优化模型,取得了良好的效果. 后续也可以将该算法应用到折扣背包问题、动静态背包问题、机器人路线规划、油田开采模型优化等的实际工作中,使其发挥更大更广泛的实际应用价值.

[参 考 文 献]

[1] MATHEWS G B. On the partition of numbers [J]. Proceedings of the London Mathematical Society, 1896, s1-28(1): 486-490.
[2] 邹恒明. 算法之道 [M]. 北京: 机械工业出版社, 2010: 67-72.
[3] KELLERER H, PFERSCHY U, PISINGER D. Knapsack Problems [M]. Berlin: Springer-Verlag, 2004.
[4] KARP R M, MILLER R E, THATCHER J W. Reducibility among combinatorial problems [J]. Journal of Symbolic Logic, 1975, 40(4): 618-619.
[5] MARTELLO S, TOTH P. Knapsack Problems: Algorithms and Computer Implementations [M]. New York: John Wiley & Sons, Inc., 1990.
[6] 王熙照, 贺毅朝. 求解背包问题的演化算法 [J]. 软件学报, 2017, 28(1): 1-16.
[7] BANSAL J C, DEEP K. A modified binary particle swarm optimization for knapsack problems [J]. Applied Mathematics and Computation, 2012, 218(22): 11042-11061.
[8] EZUGWU A E, PILLAY V, HIRASEN D, et al. A comparative study of meta-heuristic optimization algorithms for 0-1 knapsack problem: Some initial results [J]. IEEE Access, 2019(7): 43979-44001.
[9] HU J S, CHEN G L, GUO G C. Solving the 0-1 knapsack problem on quantum computer [J]. Chinese Journal of Computers, 1999, 22(12): 1314-1316.
[10] Cormen T H, Leiserson C E, Rivest R L, et al. Introduction to Algorithms [M]. 2nd ed. Cambridge: MIT Press, 2001: 323-399.
[11] PUSHPA S K, MRUNAl T V, SUHAS C. A study of performance analysis on knapsack problem [J]. International Journal of Computer Applications, 2016(2): 5-10.
[12] GOLDBERG D E. Genetic Algorithms in Search, Optimization and Machine Learning [M]. Boston: Addison-Wesley Longman Publishing Co., Inc., 1989.
[13] 霍红卫, 许进, 保铮. 基于遗传算法的0-1背包问题求解 [J]. 西安电子科技大学学报, 1999, 26(4): 101-105.
[14] 陈国良, 王熙法, 庄镇泉. 遗传算法及其应用 [M]. 北京: 人民邮电出版社, 1999: 1-195.
[15] 贺毅朝, 王熙照, 李文斌, 等. 基于遗传算法求解折扣{0-1}背包问题的研究 [J]. 计算机学报, 2016, 39(12): 2614-2630.
[16] 徐小平, 庞润娟, 王峰, 等. 求解0-1背包问题的烟花算法 [J]. 计算机系统应用, 2019, 28(2): 164-170.
[17] KENNEDY J, EBERHART R. Particle swarm optimization [C] // Proceedings of ICNN'95 - International Conference on Neural Networks. IEEE, 1995: 1942-1948.
[18] EBERHART R C, SHI Y H. Particle swarm optimization: developments, applications and resources [C]// Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No.01TH8546). IEEE, 2001: 81-86.
[19] 赵传信, 季一木. 粒子群优化算法在0/1背包问题的应用 [J]. 微机发展, 2005, 15(10): 23-25.
[20] 贺毅朝, 王熙照, 李文斌, 等. 求解随机时变背包问题的精确算法与进化算法 [J]. 软件学报, 2017, 28(2): 185-202.
[21] 贺毅朝, 刘坤起, 张翠军, 等. 求解背包问题的贪心遗传算法及其应用 [J]. 计算机工程与设计, 2007, 28(11): 2655-2657, 2681.
[22] 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.
[23] 任静敏, 潘大志. 带权重的贪心萤火虫算法求解0-1背包问题 [J]. 计算机与现代化, 2019(5): 86-91.
[24] 耿亚, 吴访升. 基于粒子群-模拟退火算法的背包问题研究 [J]. 控制工程, 2019, 26(5): 991-996.
[25] 周洋, 潘大志. 求解0-1背包问题的贪心优化粒子群算法 [J]. 西华师范大学学报(自然科学版), 2018, 39(3): 319-324.
[26] 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. IEEE, 1997: 4104-4108.
[27] KIRKPATRICK S, GELATT C D, VECCHI M P. Optimization by simulated annealing [J]. Science, 1983, 220(4598): 671-680.
[28] 卢宇婷, 林禹攸, 彭乔姿, 等. 模拟退火算法改进综述及参数探究 [J]. 大学数学, 2015, 31(06): 96-103.
[29] 庞峰. 模拟退火算法的原理及算法在优化问题上的应用 [D]. 长春: 吉林大学, 2006: 1-5
[30] 谷鹏, 王大龙, 张世仓. 基于粒子群优化的扩展卡尔曼滤波方法研究 [J]. 工业控制计算机, 2019, 32(11): 80-82.
[31] 李鹏, 李兵舰, 亓亮, 等. 一种改进的粒子群优化算法及其在无人机航路规划中的应用 [J]. 舰船电子对抗, 2019, 42(5): 59-64.
[32] 郭玉洁, 张强, 袁和平. 一种双种群协同多目标粒子群优化算法及应用 [J]. 吉林大学学报(理学版), 2019, 57(5): 1155-1162.

(责任编辑: 李 艺)