DOI：10.13718/j.cnki.xsxb.2022.11.009

改进贪心算法求解扩展简化折扣{0-1}背包问题①

林洪，　邓艳

中国人民武装警察部队警官学院 基础部，成都 610213

摘要：扩展简化折扣{0-1}背包问题(ESD{0-1}KP)是折扣{0-1}背包问题(D{0-1}KP)的拓展. ESD{0-1}KP 增加了 D{0-1}KP 中单个项集中的物品数量，导致其求解难度增加，并且现有贪心策略算子(GSOR) 算法效果不理想. 基于 ESD{0-1}KP 模型，在每个项集中增加一个价值为 0，质量为 0 的虚拟物品，同时对 ESD{0-1}KP 模型中的约束进行松弛，从理论上证明了 ESD{0-1}KP 与多选择背包问题(MCKP)等价. 结合改进帕累托算法(IPA)，提出新的贪心策略算子(NGSOR). NGSOR 首先将同一项集多个物品的选择情况通过在项集内增加物品来表示，按从价值密度从高到低顺序选择物品，若被选择物品的价值比物品所在项集已选择物品的价值更大，则对该项集进行迭代. 仿真实验结果表明：NGSOR 相比于 GSOR，求解精度平均提升 24.56%，求解速度平均提升 44.95%.

关　键　词：贪心算法；扩展折扣{0-1}背包问题(ESD{0-1}KP)；改进帕累托算法(IPA)；价值密度；多选择背包问题(MCKP)

中图分类号：TP18　　**文献标志码**：A　　**文章编号**：1000－5471(2022)11－0063－09

Improved Greedy Algorithm to Solve Extended Simplified Discounted {0-1} Knapsack Problem

LIN Hong,　DENG Yan

Department of Basic Courses, Officers College of PAP, Chengdu 610213, China

Abstract: The Extended Simplified Discount {0-1} Knapsack Problem (ESD{0-1}KP) is an extension of the Discounted {0-1} Knapsack Problem (D{0-1}KP). The ESD{0-1}KP increases the number of items in each item set in D{0-1}KP, which makes it more difficult to solve, and the existing Greedy Strategy Operator (GSOR) algorithm is not effective. Based on the ESD{0-1}KP model, a virtual item that value and weight of 0 is added to each item set, and the constraints in the ESD{0-1}KP model is relaxed. Moreover, we prove that the ESD{0-1}KP is equivalent to the Multiple Choice Knapsack Problem (MCKP) theoretically. A New Greedy Strategy Operator (NGSOR) is proposed by combining with the Improved Pareto Algorithm (IPA). NGSOR firstly expresses the selection of multiple items in the same item set by adding items, and then selects items in order of value density from highest to lowest. If the value of the selected item is more than the value of the selected item in the item set where the item is located, the item set is iterated. The simulation results show that, compared with GSOR, the NGSOR has an average improvement of 24.56% in solution accuracy and 44.95% in solution speed.

Key words: greedy algorithm; extended discounted {0-1} knapsack problem (ESD{0-1}KP); improved pareto algorithm(IPA); value density; multiple-choice knapsack problem (MCKP)

① 收稿日期：2021－10－30

作者简介：林洪，硕士研究生，助教，主要从事形式概念分析，决策优化等研究.

折扣{0-1}背包问题(discounted {0-1} knapsack problem, D{0-1}KP)作为{0-1}背包问题的拓展[1-7],因其能刻画物品选择与否对其他物品的影响关系而受到广泛关注. D{0-1}KP在项目决策、投资与预算控制等方面具有广阔的应用背景[5,7].

传统D{0-1}KP模型每个项集中物品仅有两个物品,通过在项集中增加一个物品来表示项集中两个物品同时被选择的情况. 当同一项集中物品个数较多时,传统D{0-1}KP较难投入应用.

扩展简化折扣{0-1}背包问题(extended simplified discounted {0-1} knapsack problem, ESD{0-1}KP)作为D{0-1}KP的拓展,相对于D{0-1}KP, ESD{0-1}KP增加了每个项集中物品的数量,且折扣关系也更加贴合实际情况.

ESD{0-1}KP利用遗传算法[3-4,7]、粒子群算法[6]、帝王蝶算法[8]等对问题进行求解,但随着单个项集中物品数量的增加,贪心修复操作难度加大,现有贪心策略算子[9](greedy strategy operator, GSOR)效果值得改进.

文章基于GSOR,通过在每个项集中增加一个价值质量均为0的虚拟物品,将ESD{0-1}KP转化为多选择背包问题[10-11](multiple-choice knapsack problem, MCKP),结合改进帕累托算法(improved pareto algorithm, IPA)[12],提出新型贪心策略算子(new greedy strategy operator, NGSOR). 利用4类ESD{0-1}KP大规模实例进行仿真,分析NGSOR性能.

同时通过数学转化,从理论上证明ESD{0-1}KP与MCKP是完全等价的,意味着传统上求解MCKP的算法均可通过NGSOR的操作进行转化求解,为D{0-1}KP和ESD{0-1}KP的后续研究奠定了基础.

当前对于ESD{0-1}KP问题的研究较少,且仅考虑了每个项集中存在3个元素,合计8种情况的约束. 随着项集规模的不断增加,相比传统D{0-1}KP, ESD{0-1}KP模型与算法得到进一步泛化. 同时,考虑到当前ESD{0-1}KP仅有项集内3个物品的数据集和已有算法结果,为了便于讨论和对比,模型与算法均考虑项集3个物品的情况.

1 ESD{0-1}KP模型

为了更加贴合实际商业模型,假设每个项集包含3个物品. 对于项集中的物品,若仅购买一个物品,则物品质量乘d_{3i-2};若购买两个物品,则对两个物品质量之和乘d_{3i-1};若项集中物品均被购买,则对其质量之和乘d_{3i}. 折扣系数满足:

$$0 < d_{3i-j} \leqslant 1,\ j = 0,1,2 \tag{1}$$

$$d_{3i-2} > d_{3i-1} > d_{3i} \tag{2}$$

为便于讨论,记

$$\boldsymbol{D}_i = [d_{3i-2},\ d_{3i-1},\ d_{3i}] \tag{3}$$

记问题的决策变量为$\boldsymbol{X} = [x_1, x_2, \cdots, x_{3n}] \in \{0, 1\}^{3n}$. 将决策变量中已选择的物品记为集合$K$. 对于$\forall i \in N$,若

$$x_i = 1 \tag{4}$$

则$i \in K$,且$K_j = \{i \mid i \in K,\ 3j-2 \leqslant i \leqslant 3j\}$.

将项集中物品所有选择情况记为矩阵$\boldsymbol{A}$. 则

$$\boldsymbol{A}^{\mathrm{T}} = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix} \tag{5}$$

定义1[9] 假设问题包含n个项集,每个项集3个物品,则共包含$3n$个物品,每个项集对应的3个物品的价值系数为$\boldsymbol{P} = [\boldsymbol{P}_1, \boldsymbol{P}_2, \cdots, \boldsymbol{P}_n]$,质量系数为$\boldsymbol{W} = [\boldsymbol{W}_1, \boldsymbol{W}_2, \cdots, \boldsymbol{W}_n]$,每个项集对应的折扣率为$\boldsymbol{D} = [\boldsymbol{D}_1, \boldsymbol{D}_2, \cdots, \boldsymbol{D}_n]$. 其中,$\boldsymbol{P}_i = [p_{3i-2}, p_{3i-1}, p_{3i}]$,$\boldsymbol{W}_i = [w_{3i-2}, w_{3i-1}, w_{3i}]$,$\boldsymbol{D}_i = [d_{3i-2}, d_{3i-1}, d_{3i}]$.

给定背包的最大载重为C,为了避免所有解均为可行解,因此

$$\sum_{i=1}^{n}\max(\boldsymbol{A}\times\boldsymbol{W}_i^{\mathrm{T}}\times\boldsymbol{B}_i^{\mathrm{T}})>C \tag{6}$$

则 ESD{0-1}KP 模型可描述如下：

$$\max\ f(\boldsymbol{X})=\sum_{i=1}^{3n}x_i p_i \tag{7}$$

$$\text{s. t.}\ \sum_{j=1}^{n}\Big(\sum_{k\in K_j}w_k\times d_{3j-3+\mathrm{card}(K_j)}\Big)\leqslant C \tag{8}$$

$$x_i\in\{0,1\},\ i=1,2,\cdots,n \tag{9}$$

决策变量 $x_i(1\leqslant i\leqslant 3n)$ 表示第 i 个物品是否被装入背包. 式(7)表示目标函数，即满足约束条件下获得收益最大. 式(8)通过先求解第 j 个项集中物品经过折扣之后的值，再对所有项集的值进行求和. 式(9)表示 0-1 约束.

2　传统贪心策略算子

与 D{0-1}KP 类似，ESD{0-1}KP 通过在项集中增加物品以表示项集中多个物品同时被选择的情况. 因此，对于 n 个项集，且每个项集中有且仅有 3 个物品而言，一共有 $(2^3-1)n=7n$ 种情况存在物品被选择. 记所有的物品选择情况为矩阵 $\boldsymbol{R}_{7n,3n}=\{r_{ij}\}_{7n,3n}$，且有

$$\boldsymbol{R}_{7^n,3n}=\begin{bmatrix}\boldsymbol{A}&0&\cdots&0\\0&\boldsymbol{A}&\cdots&0\\\vdots&\vdots&&\vdots\\0&0&\cdots&\boldsymbol{A}\end{bmatrix} \tag{10}$$

$\boldsymbol{R}_i(1\leqslant i\leqslant 7n)$ 表示矩阵 $\boldsymbol{R}$ 的第 i 行，则第 i 行选择情况的价值密度为

$$e_i=\frac{\boldsymbol{R}_i\times\boldsymbol{P}^{\mathrm{T}}}{(\boldsymbol{R}_i\times\boldsymbol{W}^{\mathrm{T}})\times d_{\lfloor\frac{i-1}{7}\rfloor\times 3+\sum_{j=1}^{3n}r_{ij}}} \tag{11}$$

将物品序号按照计算完成后的价值密度从高到低的顺序存储在向量 $\boldsymbol{H}$ 中. 也即对于 $\forall\,1\leqslant i<j\leqslant 7^n$，必然有 $e_{\boldsymbol{H}_i}\geqslant e_{\boldsymbol{H}_j}$.

结合公式(10)，(11)以及文献[9]提出 GSOR，算法伪代码描述见算法 1.

算法 1 GSOR

输入：$\boldsymbol{X}$

输出：$\boldsymbol{X}$

1. $T_1=\boldsymbol{X}\times\boldsymbol{P}^{\mathrm{T}}$；$T_2=0$
2. FOR $i=1$：n
3. $\quad T_2=T_2+\sum_{j=3i-2}^{3i}x_j w_j d_{3i-3+\sum_{j=3i-2}^{3i}x_j}$
4. END
5. FOR $i=1$：$7n$
6. $\quad \boldsymbol{X}'=\mathrm{sgn}(\boldsymbol{X}+\boldsymbol{R}_{\boldsymbol{H}_i})$
7. $\quad T_3=\boldsymbol{X}'\times\boldsymbol{P}^{\mathrm{T}}$；$T_4=0$
8. $\quad$ FOR $j=1$：n
9. $\quad\quad T_4=T_4+\sum_{k=3j-2}^{3j}x_k w_k d_{3j-3+\sum_{k=3j-2}^{3j}x_k}$
10. $\quad$ END
11. $\quad$ IF $T_4\leqslant C$
12. $\quad\quad$ IF $T_3>T_2$
13. $\quad\quad\quad \boldsymbol{X}=\boldsymbol{X}'$

14. END

15. END

16. END

在算法GSOR中，步1计算原有决策变量的质量，时间复杂度为$O(n)$. 步2－4计算决策变量的质量，时间复杂度为$O(n)$. 步5－16进行GSOR贪心操作，时间复杂度为$O(n^2)$. 其中，步6计算价值密度，步7计算贪心选择的新决策变量的价值，步8－10计算新决策变量的质量，步11－15选择满足约束且价值更大的决策变量进行迭代. GSOR总的时间复杂度为$O(n^2)$.

当同一项集中两种选择情况冲突时，GSOR直接对两种物品选择情况取并集.

针对MCKP同一项集中多个物品同时被选择的常见情况，IPA算法有非常良好的求解性能，且理论证明完善. 与ESD{0-1}KP中“每个项集中至多选择一项物品”约束不同，MCKP的约束为“每个项集中选择且仅选择一项物品”. 显然直接使用IPA并不可行，需要建立ESD{0-1}KP与MCKP的等价关系，再将IPA引入到ESD{0-1}KP的求解算法中，达到改进算法的目的.

3 新型贪心策略算子

为便于表述GSOR，将定义1中的ESD{0-1}KP模型转化为传统的D{0-1}KP模型，模型可表述为

$$\max f(\boldsymbol{Y})=\sum_{i=1}^{7n}y_i p'_i \tag{12}$$

$$\text{s. t. } g_1(\boldsymbol{Y})=\sum_{i=1}^{7n}y_i w'_i\leqslant C \tag{13}$$

$$h_1(\boldsymbol{Y})=\sum_{i=7j-6}^{7j}y_i\leqslant 1 \tag{14}$$

$$y_i\in\{0,1\},\ i=1,2,\cdots,7n \tag{15}$$

其中：

$$p'_i=\boldsymbol{R}_i\times\boldsymbol{P}^{\mathrm{T}} \tag{16}$$

$$w'_i=(\boldsymbol{R}_i\times\boldsymbol{W}^{\mathrm{T}})\times d_{\lfloor\frac{i}{7}\rfloor\times 3}+\sum_{j=1}^{3n}r_{ij} \tag{17}$$

针对D{0-1}KP模型，每个项集中增加一个虚拟物品，其质量与价值均为0，价值密度为0. 则模型可进一步转化如下：

$$\max f(\boldsymbol{Y})=\sum_{i=1}^{8n}y_i p'_i \tag{18}$$

$$\text{s. t. } g_2(\boldsymbol{Y})=\sum_{i=1}^{8n}y_i w'_i\leqslant C \tag{19}$$

$$h_2(\boldsymbol{Y})=\sum_{i=8j-7}^{8j}y_i=1 \tag{20}$$

$$y_i\in\{0,1\},\ i=1,2,\cdots,8n \tag{21}$$

显然，公式(20)要求在每个项集中选择且仅选择一个方案，这类约束为多选择约束. 此时问题为MCKP模型，则使用IPA对同一项集中多个物体同时被选择的情况进行求解. 下面证明ESD{0-1}KP模型与MCKP模型等价.

值得注意的是，MCKP作为一个经典的NP问题，若项集中的物品价值与质量均相同，则问题退化为KP，因此，在MCKP的假设中任意项集不存在两个物品的价值和质量均相等.

定理1 若$\boldsymbol{Y}_1$满足f_1,g_1,h_1约束，$\boldsymbol{Y}_2$满足f_2,g_2,h_2约束. 则$\boldsymbol{Y}_1$与$\boldsymbol{Y}_2$必然一一对应.

证 对于$\boldsymbol{Y}_1=[y_{11},y_{12},\cdots,y_{1,7n}]$，新增序号$7n+1$到$8n$共计$n$个物品，其物品价值与质量均为0，则有

$$f_1(\boldsymbol{Y}_1)=\sum_{i=1}^{7n}y_{1i}p'_i=\sum_{i=1}^{7n}y_{1i}p'_i+\sum_{i=7n+1}^{8n}y_{1i}\times 0=\sum_{i=1}^{8n}y_{1i}p'_i=f_2(\boldsymbol{Y}_2)$$

$$g_1(\boldsymbol{Y}_1)=\sum_{i=1}^{7n}y_{1i}w'_i=\sum_{i=1}^{7n}y_{1i}w'_i+\sum_{i=7n+1}^{8n}y_{1i}\times 0=\sum_{i=1}^{8n}y_{1i}w'_i=g_2(\boldsymbol{Y}_2)$$

$$h_1(\boldsymbol{Y}_1)=\sum_{i=7j-6}^{7j}y_{1i}=\sum_{i=7j-6}^{7j}y_{1i}+y_{7n+j}=\sum_{i=8j-7}^{8j}y_{1i}=h_2(\boldsymbol{Y}_2)$$

因此，对于 $\forall \boldsymbol{Y}_1$，必然存在 $\boldsymbol{Y}_2$，使得

$$\boldsymbol{Y}_1=f_1^{-1}\circ f_2(\boldsymbol{Y}_2)$$

$$\boldsymbol{Y}_1=g_1^{-1}\circ g_2(\boldsymbol{Y}_2)$$

$$\boldsymbol{Y}_1=h_1^{-1}\circ h_2(\boldsymbol{Y}_2)$$

下证唯一性，利用反证法，假设 $\exists \boldsymbol{Y}_2\neq\boldsymbol{Y}_3$，且

$$\boldsymbol{Y}_1=f_1^{-1}\circ f_2(\boldsymbol{Y}_3)$$

$$\boldsymbol{Y}_1=g_1^{-1}\circ g_2(\boldsymbol{Y}_3)$$

$$\boldsymbol{Y}_1=h_1^{-1}\circ h_2(\boldsymbol{Y}_3)$$

显然，若 $h_1^{-1}\circ h_2(\boldsymbol{Y}_2)=h_1^{-1}\circ h_2(\boldsymbol{Y}_3)$，则必然存在某一项集中有两个质量与价值相等的物品，与假设不符. 故原命题成立.

综上，结论成立.

MCKP 的相关内容可参考文献[10-14].

通过增加一个虚拟物品使得ESD{0-1}KP和MCKP在数学模型上完全一致，即传统意义上求解MCKP的算法都能够通过类似增加虚拟物品的操作，对ESD{0-1}KP和D{0-1}KP进行求解，丰富了ESD{0-1}KP和D{0-1}KP求解算法.

从定理1中不难发现，ESD{0-1}KP与MCKP是完全等价的，因此，对于求解MCKP问题的优秀算法也可以用于求解ESD{0-1}KP.

ESD{0-1}KP可采用MCKP的贪心策略进行处理. 同时，公式(18)－(21)模型仍存在传统D{0-1}KP模型缺陷，即非正常个体编码较多，可将MCKP中的贪心策略应用到ESD{0-1}KP模型中.

因IPA拥有计算速度快、求解精度高和算法结构简单等特点，因此基于IPA提出适用于ESD{0-1}KP的NGSOR.

记第 i 个项集的所有物品为 N_i，个数为 n. 第 i 个项集中所有的线性非支配(linear programming undominated，LP-undominated)物品集合称线性非支配子集，并记为 L_i. 由文献[15]可知：

定理 2[10]　对于 $\forall s,t\in N_i$，$w_{is}>w_{ir}$，$w_{it}>w_{ir}$，$p_{it}>p_{ir}$，若 $e_{it}>e_{is}$，且 $p_{it}\geqslant p_{is}$，则 $s\notin L_i$.

在式(18)－(21)模型中，结合定理2可知，$w_{is}>w_{ir}=0$，$w_{it}>w_{ir}=0$，$p_{it}>p_{ir}=0$. 对于同一项集中两个物品 s,t 同时被选择时，若 $e_{it}>e_{is}$，且 $p_{it}\geqslant p_{is}$，则选择物品 t 而不选择物品 s.

即对于ESD{0-1}KP，当按照选择方案的价值密度从高到低选择时，若同一项集中两个选择方案同时被选择，应当选择价值更大的选择方案. 由此得到NGSOR算法，算法伪代码如下：

算法 2　NGSOR

输入：$\boldsymbol{X}$

输出：$\boldsymbol{X}$

1. FOR $i=1:n$

2. $\boldsymbol{P}_s=[p_{3i-2},\ p_{3i-1},\ p_{3i-2}+p_{3i-1},\ p_{3i},\ p_{3i-2}+p_{3i},\ p_{3i-1}+p_{3i},\ p_{3i-2}+p_{3i-1}+p_{3i}]$

3. $\boldsymbol{W}_s=[d_{3i-2}w_{3i-2},\ d_{3i-2}w_{3i-1},\ d_{3i-1}(w_{3i-2}+w_{3i-1}),\ d_{3i-2}w_{3i},\ d_{3i-1}(w_{3i-2}+w_{3i}),\ d_{3i-1}(w_{3i-1}+w_{3i}),\ d_{3i-2}(w_{3i-2}+w_{3i-1}+w_{3i})]$

4. $\boldsymbol{P}'=[\boldsymbol{P}',\ \boldsymbol{P}_s]$; $\boldsymbol{W}'=[\boldsymbol{W}',\ \boldsymbol{W}_s]$

5. END

6. $\boldsymbol{E}=\boldsymbol{P}'./\boldsymbol{W}'$; $\boldsymbol{E}=[\boldsymbol{E};\ 1:7n]$

7. $\boldsymbol{E}=-\mathrm{sortrows}(-\boldsymbol{E}',\ 1)'$; $\boldsymbol{H}=\boldsymbol{E}_2$

8. $\boldsymbol{X}=\mathrm{zeros}(1,\ n)$

9. FOR $i=1:7n$
10. $t_1=\lfloor\frac{\boldsymbol{H}_i-1}{7}\rfloor+1$
11. $t_2=\boldsymbol{X}_{3t_1-2:3t_1}\times\boldsymbol{P}^{\mathrm{T}}_{3t_1-2:3t_1}$
12. IF $t_2=0$
13. $e_1=0$
14. ELSE
15. $t_3=w'_{X_{3t_1-2:3t_1}\times[1,2,4]'+3t_1-3}$; $e_1=t_2/t_3$
16. END
17. $t_4=p'_{\boldsymbol{H}_i}$; $t_5=w'_{\boldsymbol{H}_i}$; $e_2=t_4/t_5$
18. $\boldsymbol{T}=\boldsymbol{X}$; $\boldsymbol{T}_{3t_1-2:3t_1}=\boldsymbol{A}_{\boldsymbol{H}_i-7t_1+7}$; $Q=0$
19. FOR $j=1:n$
20. IF $\mathrm{sum}(\boldsymbol{T}_{3j-2:3j})>0$
21. $Q=Q+w'_{\boldsymbol{T}_{3j-2:3j}\times[1,2,4]'+7j-7}$
22. END
23. END
24. IF $Q\leqslant C$
25. IF$t_2<t_4$
26. $\boldsymbol{X}=\boldsymbol{T}$
27. END
28. END
29. END

NGSOR与GSOR在时间复杂度上无数量级差异，但在GSOR中的步3与步9都需要重新计算每个项集选择情况所对应的质量，而在NGSOR中则通过存储后直接调用，因此NGSOR算法求解速度更快.

4 实例计算与结果对比

为充分展示NGSOR的算法性能，不仅与现有GSOR进行对比，同时与将文献[9]贪心遗传算法(greedy genetic algorithm，GGA)中的GSOR替换成NGSOR后得到的新型贪心遗传算法(new greedy genetic algorithm，NGGA)进行对比. 关于遗传算法的相关内容可参考文献[16－18].

因NGGA仅将GGA中的贪心策略进行了修改，因此沿用文献[9]中的参数算法设种群为50，最大迭代次数为100次，交叉概率$p_c=0.8$，变异概率$p_m=0.01$. 此外，ESD{0-1}KP中，折扣系数为$d_{3i-2}=1$，$d_{3i-1}=0.8$，$d_{3i}=0.7$.

测试数据集沿用文献[9]中4类大规模实例，具体测试数据参数可参考文献[9]. 其中，数据规模为$300\leqslant n\leqslant 3\,000$，且

1) EUDKP1-EUDKP10，参数设置$w_j\in{}_z[2,1\,000]$，$p_j\in{}_z[2,1\,000]$.

2) EWDKP1-EWDKP10，参数设置$w_j\in{}_z[101,1\,000]$，$p_j\in{}_z[w_j-100,w_j+100]$.

3) ESDKP1-ESDKP10，参数设置$w_j\in{}_z[2,1\,000]$，$p_j\in w_j+100$.

4) EIDKP1-EIDKP10，参数设置$p_j\in{}_z[2,1\,000]$，$w_j\in p_j+100$.

其中${}_z[a,b]$表示在整数a到整数b之间的整数集.

文章使用计算机基本配置为：Intel Core i7-8700 CPU@3.2GHz(12核)，16GB DDR4L，Microsoft Windows 10家庭版. 利用MATLAB R2016a对问题进行求解及绘图.

4.1 求解结果

利用NGSOR，GSOR，GGA和NGGA对4类数据进行求解. 因NGSOR和GSOR算法为非随机算法，因此仅参考最优值和求解时长指标. 对于GGA和NGGA，因其为随机算法，因此除了最优值和求解时

长以外，还需要增加平均值和最差值指标.

为避免单次计算的偶然因素干扰，所有算法的求解时长为 30 次独立求解均值. 同时，为检验 NGSOR 的算法效果，对 GGA 重新进行测试.

算法计算结果见表 1，其中折扣背包问题的动态规划算法(dynamic programming of discounted knapsack problem，DPDKP)计算得到的结果为精确解.

表 1　4 类实例计算结果

实例	DPDKP	NGSOR		GSOR		NGGA				GGA			
		最优值	时长/s	最优值	时长/s	最优值	平均值	最差值	时长/s	最优值	平均值	最差值	时长/s
EUDKP1	127 113	121 314	0.09	73 287	0.14	122 077	121 725	121 507	1.11	84 371	82 538	81 251	1.13
EUDKP2	265 790	255 241	0.34	147 215	0.54	255 874	255 468	255 241	2.10	171 338	167 225	163 865	2.27
EUDKP3	387 730	369 604	0.74	202 087	1.23	370 048	369 669	369 604	3.18	233 371	229 068	226 370	3.46
EUDKP4	548 114	522 328	1.33	327 590	2.22	522 629	522 357	522 328	4.28	348 634	343 995	339 959	4.72
EUDKP5	605 824	568 926	2.02	299 638	3.58	569 293	568 941	568 926	5.34	355 598	346 993	339 347	6.07
EUDKP6	759 493	720 552	2.91	413 396	5.04	720 552	720 552	720 552	6.51	477 962	463 119	444 117	7.66
EUDKP7	895 538	849 585	3.95	450 381	6.73	849 585	849 585	849 585	7.71	541 906	534 008	525 024	9.21
EUDKP8	1 014 189	964 946	5.17	528 649	8.93	964 946	964 946	964 946	9.61	622 543	615 602	608 823	10.71
EUDKP9	1 177 138	1 118 913	6.52	645 551	11.36	1 118 913	1 118 913	1 118 913	11.50	723 176	711 243	700 019	12.81
EUDKP10	1 244 094	1 181 595	7.95	630 010	13.78	1 181 595	1 181 595	1 181 595	12.72	748 864	727 854	707 139	14.03
EWDKP1	81 559	81 552	0.07	59 902	0.14	81 552	81 552	81 552	0.94	69 477	68 448	66 926	1.24
EWDKP2	208 195	207 976	0.31	176 627	0.55	207 976	207 976	207 976	1.98	176 627	176 627	176 627	2.25
EWDKP3	258 117	257 555	0.66	167 679	1.26	257 555	257 555	257 555	3.02	202 132	195 898	190 019	3.62
EWDKP4	424 542	424 259	1.24	379 883	2.20	424 259	424 259	424 259	4.36	379 883	379 883	379 883	4.86
EWDKP5	492 351	492 206	1.87	395 913	3.50	492 206	492 206	492 206	5.43	395 913	395 913	395 913	6.17
EWDKP6	573 930	573 582	2.60	450 306	5.13	573 582	573 582	573 582	6.31	485 008	478 767	472 090	7.90
EWDKP7	849 993	849 624	3.92	762 266	6.74	849 624	849 624	849 624	8.23	762 266	762 266	762 266	9.47
EWDKP8	841 881	841 427	4.97	709 915	8.83	841 427	841 427	841 427	9.37	729 637	726 186	723 637	10.96
EWDKP9	886 216	885 812	5.90	680 928	11.37	885 812	885 812	885 812	10.27	762 121	736 644	720 255	13.18
EWDKP10	1 098 642	1 097 735	7.81	863 944	13.97	1 097 735	1 097 735	1 097 735	12.49	904 869	896 207	889 067	14.81
ESDKP1	113 829	113 412	0.08	96 293	0.14	113 536	113 421	113 412	1.08	96 293	96 293	96 293	1.16
ESDKP2	187 048	186 274	0.31	154 219	0.56	186 352	186 290	186 274	2.00	154 219	154 219	154 219	2.08
ESDKP3	303 535	302 825	0.69	253 952	1.24	302 825	302 825	302 825	2.96	253 952	253 952	253 952	3.30
ESDKP4	435 400	434 875	1.24	368 468	2.22	434 875	434 875	434 875	4.28	368 468	368 468	368 468	4.62
ESDKP5	522 779	521 506	1.97	444 411	3.48	521 506	521 506	521 506	5.50	444 411	444 411	444 411	5.99
ESDKP6	647 891	646 919	2.83	556 868	5.07	646 919	646 919	646 919	6.72	556 868	556 868	556 868	7.37
ESDKP7	781 520	780 157	3.83	669 512	6.91	780 157	780 157	780 157	8.04	669 512	669 512	669 512	8.84
ESDKP8	837 800	836 120	4.87	706 367	8.89	836 120	836 120	836 120	9.30	706 367	706 367	706 367	10.26
ESDKP9	918 049	916 055	6.20	769 091	11.48	916 055	916 055	916 055	10.87	769 091	769 091	769 091	12.03
ESDKP10	1 231 198	1 229 645	8.01	1 071 431	14.23	1 229 645	1 229 645	1 229 645	12.66	1 071 431	1 071 431	1 071 431	14.34
EIDKP1	81 492	81 446	0.07	64 552	0.14	81 446	81 446	81 446	0.91	76 870	74 476	72 388	1.16
EIDKP2	182 428	182 421	0.27	148 323	0.56	182 421	182 421	182 421	1.76	171 079	164 968	159 511	2.30
EIDKP3	243 972	243 971	0.60	193 672	1.25	243 971	243 971	243 971	2.61	225 393	219 971	214 864	3.53
EIDKP4	107 965	107 965	1.08	76 248	2.22	107 965	107 965	107 965	3.78	97 501	95 082	91 187	4.80
EIDKP5	168 288	168 211	1.87	131 167	3.55	168 211	168 211	168 211	5.26	136 111	134 876	134 157	6.37
EIDKP6	211 004	210 908	2.80	170 650	4.94	210 908	210 908	210 908	6.59	173 301	172 822	172 500	8.00
EIDKP7	249 009	248 971	3.75	200 691	6.87	248 971	248 971	248 971	7.84	203 294	203 074	202 710	9.51
EIDKP8	246 243	246 189	4.33	185 199	8.84	246 189	246 189	246 189	8.62	194 100	192 687	191 251	10.95
EIDKP9	282 318	282 292	5.67	212 532	11.31	282 292	282 292	282 292	10.31	221 640	218 988	217 082	12.97
EIDKP10	351 482	351 422	7.35	280 593	13.94	351 422	351422	351 422	12.00	285 983	283 731	282 675	14.87

4.2 结果分析

由表1知，基于EUDKP算例，相比于GSOR的算法精度，NGSOR的误差范围从40.23%～50.54%缩小至3.97%～6.09%，对所有算例的提升精确度进行平均计算得到算法精度平均提升41.42%，同理得到算法速度平均提升40.68%.应用到遗传算法中，NGGA相比于GGA，最优解精度平均提升33.22%，平均值精度平均提升34.42%.

基于EWDKP算例，相比于GSOR的算法精度，NGSOR的误差范围从10.32%～35.04%缩小至0.01%～0.22%，算法精度平均提升19.82%，同时算法速度平均提升45.85%.应用到遗传算法中，NGGA相比于GGA，最优解精度平均提升15.18%，平均值精度平均提升16.07%.

基于ESDKP算例，相比于GSOR的算法精度，NGSOR的误差范围从12.98%～17.55%缩小至0.12%～0.41%，算法精度平均提升15.07%，同时算法速度平均提升44.31%.应用到遗传算法中，NGGA相比于GGA，最优解精度平均提升15.08%，平均值精度平均提升15.07%.

基于EIDKP算例，相比于GSOR的算法精度，NGSOR的误差范围从18.70%～29.38%缩小至0.00%～0.06%，算法精度平均提升21.97%，同时算法速度平均提升48.94%.应用到遗传算法中，NGGA相比于GGA，最优解精度平均提升14.56%，平均值精度平均提升15.96%.

整体分析，就算法精度讨论，NGSOR平均误差为1.31%，GSOR平均误差为25.87%，平均提升24.56%.就算法速度而言，平均提升44.95%.总体上，NGSOR相对于GSOR具有明显优势，更加适应用于ESD{0-1}KP的求解，效果理想.

为了更好展示NGSOR性能，将表1数据绘制成图1与图2.从图1中不难发现，NGSOR与NGGA求解效果均比GSOR和GGA效果更好.从图2中可以知道，在求解时长方面，NGGA比GGA更快，且NGSOR也比GSOR更快.

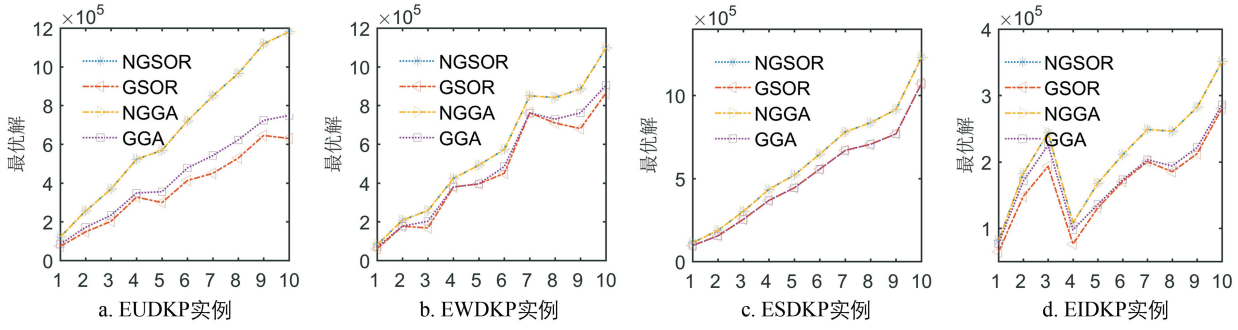

图1 最优解

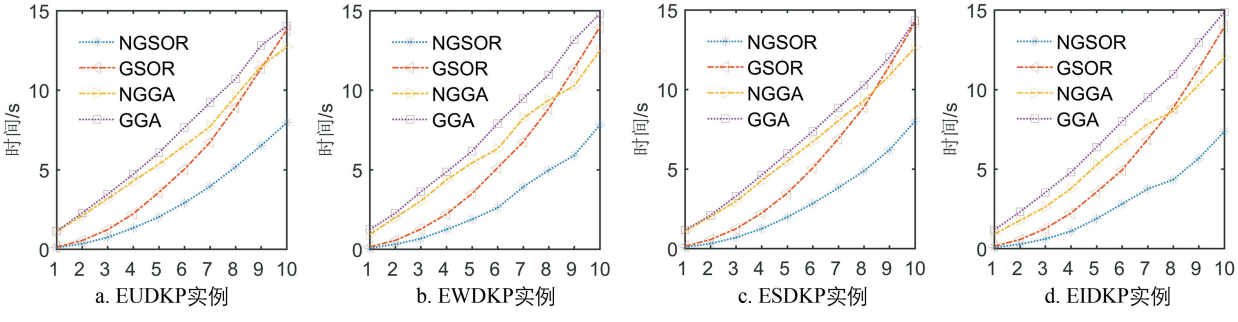

图2 求解时长

5 结束语

文章基于GSOR，通过改进模型，将ESD{0-1}KP与D{0-1}KP转化为MCKP，再引入IPA对问题进行处理，算法性能提升明显.值得注意的是，当ESD{0-1}KP的项集中物品数量继续增加时，无论是NGSOR还是GSOR都需要指数级储存空间，NGSOR虽然能够提供更好的贪心结果，但其内存需求与GSOR并无特别大的改进.下一步工作，不妨思考新的支配关系与剪枝策略，提出更加优秀的贪心算法.

参考文献：

[1] GUDER J. Discounted Knapsack Problem for Pairs of Items [D]. Nuremberg：University of Erlangen-Nurnberg，2005.

[2] GULDAN B. Heuristic and Exact Algorithms for Discounted Knapsack Problems [D]. Nuremberg：University of Erlangen-Nurnberg，2007.

[3] 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.

[4] 杨洋，潘大志，刘益，等. 折扣{0-1}背包问题的简化新模型及遗传算法求解 [J]. 计算机应用，2019，39(3)：656-662.

[5] RONG A Y，FIGUEIRA J R，KLAMROTH K. Dynamic Programming Based Algorithms for the Discounted {0-1} Knapsack Problem [J]. Applied Mathematics and Computation，2012，218(12)：6921-6933.

[6] HE Y C，WANG X Z，HE Y L，et al. Exact and Approximate Algorithms for Discounted {0-1} Knapsack Problem [J]. Information Sciences，2016，369：634-647.

[7] 杨洋，潘大志，贺毅朝. 改进修复策略遗传算法求解折扣{0-1}背包问题 [J]. 计算机工程与应用，2018，54(21)：37-42，132.

[8] 冯艳红，杨娟，贺毅朝，等. 差分进化帝王蝶优化算法求解折扣{0-1}背包问题 [J]. 电子学报，2018，46(6)：1343-1350.

[9] 张琴，潘大志. 扩展 SD{0-1}KP 背包问题的建模及其遗传算法求解 [J]. 西华师范大学学报(自然科学版)，2020，41(2)：214-220.

[10] NAUSS R M. The 0-1 Knapsack Problem with Multiple Choice Constraints [J]. European Journal of Operational Research，1978，2(2)：125-131.

[11] PISINGER D. A Minimal Algorithm for the Multiple-Choice Knapsack Problem [J]. European Journal of Operational Research，1995，83(2)：394-410.

[12] NAUSS R M. The 0-1 Knapsack Problem with Multiple Choice Constraints [J]. European Journal of Operational Research，1978，2(2)：125-131.

[13] BALINTFY J L，ROSS G T，SINHA P，et al. A Mathematical Programming System for Preference and Compatibility Maximized Menu Planning and Scheduling [J]. Mathematical Programming，1978，15(1)：63-76.

[14] SINHA P，ZOLTNERS A A. The Multiple-Choice Knapsack Problem [J]. Operations Research，1979，27(3)：503-515.

[15] 杨洋. 改进帕累托算法求解超大规模多选择背包问题 [J]. 电子学报，2020，48(6)：1205-1212.

责任编辑　张枸