Available online at www.sciencedirect.com
ScienceDirect
Computers & Operations Research 35 (2008) 2292–2306
computers & operations research
www.elsevier.com/locate/cor

Core problems in bi-criteria {0, 1}-knapsack problems

Carlos Gomes da Silva[b,c,*], João Clímaco[a,c], José Rui Figueira[d,1]

[a] Faculdade de Economia da Universidade de Coimbra, Av. Dias da Silva, 165, 3004-512 Coimbra, Portugal
[b] Escola Superior de Tecnologia e Gestão de Leiria, Morro do Lena, Alto Vieiro, 2401-951 Leiria, Portugal
[c] INESC-Coimbra, Rua Antero de Quental, 199, 3000-033 Coimbra, Portugal
[d] CEG-IST, Center for Management Studies, Departamento de Engenharia e Gestão, Instituto Superior Técnico, Tagus Park, Av. Cavaco Silva, 2780-990 Porto Salvo, Portugal

Available online 18 January 2007

Abstract

The most efficient algorithms for solving the single-criterion {0, 1}-knapsack problem are based on the core concept (i.e., based on a small number of relevant variables). But this concept is not used in problems with more than one criterion. The main purpose of this paper is to validate the existence of such a set of variables in bi-criteria {0–1}-knapsack instances. Numerical experiments were performed on five types of {0, 1}-knapsack instances. The results are presented for the supported and non-supported solutions as well as for the entire set of efficient solutions. A description of an approximate and an exact method is also presented.

Keywords: Bi-criteria knapsack problem; Core problem; Combinatorial optimization

1. Introduction

The {0, 1}-knapsack problem consists of selecting a set of items such that their total profit is maximized and their total weights does not exceed the capacity of the knapsack. The {0, 1}-*knapsack problem* can be formulated as follows:

$$\begin{aligned} &\max \quad z(x) = z(x_1, \ldots, x_j, \ldots, x_n) = \sum_{j=1}^{n} c_j x_j \\ &\text{s.t.} \\ &\qquad \sum_{j=1}^{n} w_j x_j \leqslant W, \\ &\qquad x_j \in \{0, 1\}, \quad j = 1, \ldots, n, \end{aligned} \tag{1}$$

* Corresponding author. Escola Superior de Tecnologia e Gestão de Leiria, Morro do Lena, Alto Vieiro, 2401-951 Leiria, Portugal. Tel.: +351 244 820 300; fax: +351 244 820 301.

E-mail addresses: cgsilva@estg.ipleiria.pt (C. Gomes da Silva), jclimaco@inescc.pt (J. Clímaco), figueira@ist.utl.pt (J. Rui Figueira).

[1] Visiting Researcher (January–February, 2004) at the Operational Research Department, London School of Economics, Houghton Street, London WC2A 2AE, UK.

0305-0548/$ - see front matter
doi:10.1016/j.cor.2006.11.001

where n is the number of items, c_j is the profit of item j ($j = 1, \ldots, n$) in criterion function $z(x)$, w_j is the weight of item j, and W is the knapsack capacity. If item j is selected, $x_j = 1$; otherwise $x_j = 0$.

Dantzig [1] showed that an optimal solution for the continuous {0, 1}-knapsack problem could be obtained by sorting the items according to their non-increasing *profit-to-weight ratios* (also called *efficiencies)*, and adding them to the knapsack until the capacity is reached. In the end, there is just one item that cannot be entirely added to the knapsack. This item, b, is called the *break* or *critical item* and is defined such that $\sum_{j=1}^{b-1} w_j \leqslant W < \sum_{j=1}^{b} w_j$.

With the items ordered so that

$$\frac{c_1}{w_1} \geqslant \cdots \geqslant \frac{c_j}{w_j} \geqslant \cdots \geqslant \frac{c_n}{w_n} \tag{2}$$

an optimal solution to the continuous {0, 1}-knapsack, $\bar{x}$, also called in this paper *Dantzig solution*, is thus:

$$\bar{x}_j = \begin{cases} 1, & j < b, \\ \dfrac{W - \sum_{t=1}^{b-1} w_t}{w_b}, & j = b, \\ 0, & j > b. \end{cases} \tag{3}$$

Balas and Zemel [2] observed that for randomly generated instances an optimal solution to the {0, 1}-knapsack problem formulated in (1) is very similar to the Dantzig solution. This similarity led to the introduction of the core concept. Assuming that x^* is an optimal solution of problem (1), the *core* is $C = \{j_1, \ldots, j_2\}$, where $j_1 = \min\{j : x_j^* = 0, j = 1, \ldots, n\}$ and $j_2 = \max\{j : x_j^* = 1, j = 1, \ldots, n\}$, with the items sorted according to their non-increasing efficiencies. Please note that the order expressed in (2) is not unique when several items have the same efficiency ratio. In this case the core is one of several possible cores.

Thus, the *core* is a subset of items whose efficiencies are similar to the efficiency of the break item. This subset must be considered in order to determine an exact solution, leading to the definition of the so-called *core problem*. Results on large size instances have shown that the size of the core is quite small in relation to the total number of items, and it increases very slowly as the total number of items increases [2,25], which supports the existence of a small, but relevant problem. The core concept was the underlying concept in the development of the most efficient known algorithms for the {0, 1}-knapsack: Fayard and Plateau [3], Martello and Toth [4,5], and Pisinger [6]. The first two research teams approximated the core and set the values of all the variables outside the core equal to 1 (items with high efficiency ratios) or 0 (items with low efficiency ratios). The original knapsack problem was thus reduced to include only the items that pertained to the core. The use of the core concept evolved (see [7] for a description of the use of the core in the construction of knapsack algorithms). Pisinger [8] showed that the core could be determined while the algorithm was running, during the search for an optimal solution, thus avoiding the need to approximate the core.

In single-criterion {0, 1}-knapsack problems the concept of core is quite important because it makes it possible to avoid the complete sorting of the items required for deriving better upper and lower bounds. As Balas and Zemel [2] note, this sorting process takes up a very significant part of the total computational time. It is also important because the solution of the core problem can further be used to improve the lower bounds for an optimal solution of the original knapsack problem, thus making it possible to set a significant number of variables at their optimal value.

The concept of core is also part of the competitive algorithms used to solve important variants of the knapsack problem, such as the bounded knapsack, the equality knapsack, the multiple-choice knapsack, the subset-sum and the unbounded knapsack (see [9] for references). Nevertheless, this concept has been ignored in the studies of multiple criteria {0, 1}-knapsack problems. The main advantage of studying the core in such problems is that it focuses the search on a set of interesting items. Indeed, if the size of the core associated to the solutions of the multiple criteria {0, 1}-knapsack problem is small, it means that a significant number of those solutions can be found by just searching a small number of variables, which proves crucial to the performance of the proposed methods.

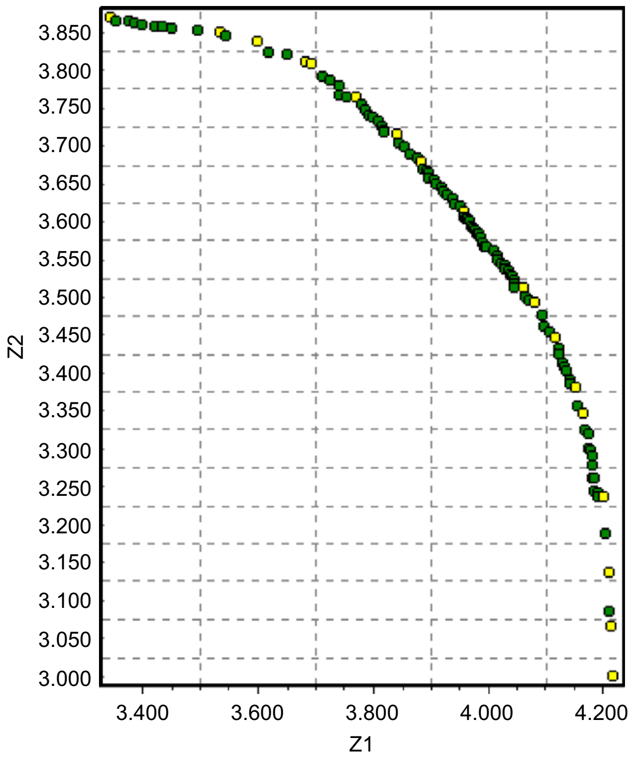

Fig. 1. Non-dominated solutions of a random instance.

The paper investigates the existence of the core structure highlighted by Balas and Zemel [2] in solutions to the bi-criteria problem:

$$
\begin{aligned}
&\max \quad z_1(x) = \sum_{j=1}^{n} c_j^1 x_j \\
&\max \quad z_2(x) = \sum_{j=1}^{n} c_j^2 x_j \\
&\text{s.t.} \\
&\qquad \sum_{j=1}^{n} w_j x_j \leqslant W, \\
&\qquad x_j \in \{0, 1\}, \quad j = 1, \ldots, n,
\end{aligned}
\tag{4}
$$

where c_j^i represents the profit of item j on criterion function $z_i(x)$, $i = 1, 2$. We assume that c_j^1, c_j^2, W, and w_j are positive integers and that $w_j \leqslant W$, $j = 1, \ldots, n$ with $\sum_{j=1}^{n} w_j > W$. Constraints $\sum_{j=1}^{n} w_j x_j \leqslant W$ and $x_j \in \{0, 1\}$, $j = 1, \ldots, n$, define the feasible region in the *decision space*, and the image of the feasible set according to the criteria functions $z_1(x)$ and $z_2(x)$ define the feasible region in the *criterion space* (i.e., the space containing the images of solutions using the criteria functions $z_1(x)$ and $z_2(x)$). A feasible *solution*, x, is said to be *efficient* if and only if there is no feasible solution, y, such that $z_i(y) \geqslant z_i(x)$, $i = 1, 2$ and $z_i(y) > z_i(x)$ for at least one i. The image of an efficient solution in the criterion space is called a *non-dominated solution*.

To solve problem (4), the set of all the efficient/non-dominated solutions must be determined. Fig. 1 shows the set of non-dominated solutions of an instance with 100 items, whose coefficients were randomly generated supposing a uniform distribution.

Certain efficient/non-dominated solutions can be obtained by maximizing the weighted-sums of the criteria, called supported efficient/non-dominated solutions. However, certain solutions, called *non-supported efficient/non-dominated* solutions, cannot be obtained in this way, because despite being efficient/non-dominated, they are convex dominated by weighted-sums of the criteria. The non-supported non-dominated solutions are located in the dual gaps of consecutive supported non-dominated solutions [10].

The solution process for bi-criteria problem could be considerably enhanced by applying the developments that have been proposed for solving single-criterion problems. In fact, problem (4) could be solved by computing the supported efficient solutions that maximize weighted-sum functions (i.e., solving single-criterion problems) and then computing the approximate solutions of single-criterion optimizations (i.e., those that approach the maximizations, just before reaching their optima). For the supported efficient solutions the available results for the single criterion {0, 1}-knapsack problem would be valid. As far as we know, no study of these non-supported efficient solutions exists in the literature.

Despite the similarities between the single-criterion problem (1) and the bi-criteria problem (4) the known algorithms for solving the second [11–13] are limited in scope compared to those proposed for solving the first, both in terms of computational time needed and size of instances that can be solved. Even the approximate methods raise too many problems related to the quality of the approximation [14–16].

The presence in the set of efficient solutions of (4) of the features reported by Balas and Zemel [2], could pave the way for the development of better approximate and exact algorithms that reduce the computational time and improve the quality of the approximation. This paper examines the possibilities offered by the similarities described above and proposes an approximate method and an exact method that take advantage of these similarities through the use of the core concept.

The rest of the paper is organized as follows: Section 2 presents the concept of bi-criteria core. Section 3 describes the computational experiments related to the size of the bi-criteria core. Section 4 describes our proposed approximate and exact methods for solving the bi-criteria {0, 1}-knapsack problem. Finally, Section 5 highlights the main conclusions that can be drawn from our research and outlines several ideas for future research.

2. Bi-criteria cores

In multiple criteria problems there is no single function. Though their criteria functions can be aggregated into a single function in several ways. In the bi-criteria case an aggregation function may be expressed by $z(x,\lambda)=\lambda z_1(x)+(1-\lambda)z_2(x)$ with $0\leqslant\lambda\leqslant 1$. Let $\Im$ be a family of weighted-sum functions $z(x,\lambda)$. We propose the following definition of core given an efficient solution x:

Definition 1. Given the family of weighted-sum functions, $\Im$, the bi-criteria core of an efficient solution x to problem (4) is the smallest core, when each function of $\Im$ is considered individually.

Thus, considering the existence of p efficient solutions and q functions, the core associated with an efficient solution x^t, taking into account the function $z(x,\lambda^k)$, is $C^{k,t}=\{j_1^{k,t},\dots,j_2^{k,t}\}$, where $j_1^{k,t}=\min\{j : x_j^t=0,\ j=1,\dots,n\}$ and $j_2^{k,t}=\max\{j : x_j^t=1,\ j=1,\dots,n\}$ (if $j_1^t>j_2^t$ we assume that $C^t=\emptyset$) and where the items are ordered according to the non-increasing values of the ratio $(\lambda^k c_j^1+(1-\lambda^k)c_j^2)/w_j,\ \ j=1,\dots,n$.

The *bi-criteria core* of x^t is $C^{k^*,t}=\arg\min_{k=1,\dots,q}\{|C^{k,t}|\}$.

According to Definition 1, determining the bi-criteria core of an efficient solution requires that all the functions $z(x,\lambda)$ of $\Im$ be analyzed. In order to obtain the smallest cores, the best function for determining the core of a given efficient solution must be identified, which means finding the value of λ that produces the smallest core.

To determine the core of an efficient solution, the items of the {0, 1}-knapsack problem must be sorted by the non-increasing values of the efficiency ratio:

$$e_j(\lambda)=\frac{\lambda c_j^1+(1-\lambda)c_j^2}{w_j}=\frac{c_j^2}{w_j}+\frac{c_j^1-c_j^2}{w_j}\lambda,\quad 0\leqslant\lambda\leqslant 1. \tag{5}$$

The efficiency ratio $e_j(\lambda)$ is a function of λ. For this reason, it is said that the efficiency ratio is not well defined. The ratio is, however, bounded from below and above according to $\min\{c_j^1/w_j, c_j^2/w_j\}\leqslant e_j(\lambda)\leqslant\max\{c_j^1/w_j, c_j^2/w_j\}$.

For a given λ', the items can be ordered such that

$$e_{l_1}(\lambda') \geqslant e_{l_2}(\lambda') \geqslant \cdots \geqslant e_{l_n}(\lambda'), \tag{6}$$

where $\{l_1, l_2, \ldots, l_n\} = \{1, 2, \ldots, n\}$.

When several items have the same efficiency ratio, the order described in (6) is not the only one possible, thus several cores are possible.

All the possible permutations of the items with the same efficiency ratio were not determined and verified, which means that the proposed cores could be even smaller.

As λ changes within [0, 1], the order defined in (6) is not stable. The order $e_{l_1}(\lambda') \geqslant \cdots \geqslant e_{l_j}(\lambda') \geqslant e_{l_{j+1}}(\lambda') \geqslant \cdots \geqslant e_{l_n}(\lambda')$, is kept constant for $\lambda^{\min} \leqslant \lambda \leqslant \lambda^{\max}$, where $\lambda^{\max}$ is provided by an optimal solution of the following linear problem:

$$\begin{aligned}
&\text{Max} \quad \lambda \\
&\text{s.t.} \\
&\qquad e_{l_j}(\lambda) \geqslant e_{l_{j+1}}(\lambda), \quad j = 1, \ldots, n-1, \\
&\qquad \lambda \leqslant 1, \quad \lambda \geqslant 0.
\end{aligned} \tag{7}$$

Using the expression of $e_{l_j}(\lambda)$, $j = 1, \ldots, n$,

$$\lambda^{\max} = \min \left\{ \frac{\dfrac{c^2_{l_{j+1}}}{w_{l_{j+1}}} - \dfrac{c^2_{l_j}}{w_{l_j}}}{\dfrac{c^1_{l_j} - c^2_{l_j}}{w_{l_j}} - \dfrac{c^1_{l_{j+1}} - c^2_{l_{j+1}}}{w_{l_{j+1}}}} : \frac{c^1_{l_j} - c^2_{l_j}}{w_{l_j}} - \frac{c^1_{l_{j+1}} - c^2_{l_{j+1}}}{w_{l_{j+1}}} < 0,\ j = 1, \ldots, n-1 \right\} \tag{8}$$

provides an optimal solution to the linear problem (7).

For a $\lambda > \lambda^{\max}$ a different ordering is defined. Thus, in order to divide the interval [0, 1] into sub-intervals $[0, u_1]$, $[u_1, u_2], \ldots, [u_{m-1}, 1]$, in which the order (6) is preserved, the following procedure is used:

Step 1: $t \leftarrow 0$, $\lambda^t \leftarrow 0$.

Step 2: Sort the items according to relation (6) with λ^t.

Step 3: Let $\lambda^{\max}$ be the optimal solution of problem (7), as defined in expression (8). If $\lambda^{\max} \geqslant 1$ then $u_{t+1} \leftarrow 1$. Once all the sub-intervals have been determined, Stop.

Step 4: $t \leftarrow t+1$, $u_{t+1} \leftarrow \lambda^{\max}$, $\lambda^t \leftarrow \lambda^{\max} + \varepsilon$ (perturbation factor, $\varepsilon > 0$). Go to Step 2.

From this procedure, the following proposition can be deduced:

Proposition 1. *The number of different orders defined as in relation* (6) *is equal to the number of sub-intervals obtained by solving sequences of problem* (7) *until* $\lambda \geqslant 1$, *starting from the order produced with* $\lambda = 0$.

Once all the sub-intervals are identified, the bi-criteria core can finally be computed.

Example 1. Let us consider the following instance of the bi-criteria {0, 1}-knapsack problem.

$$\begin{aligned}
&\max \quad z_1(x) = 85x_1 + 31x_2 + 33x_3 + 25x_4 + 28x_5 + 15x_6 + 29x_7 \\
&\max \quad z_2(x) = 72x_1 + 17x_2 + 47x_3 + 83x_4 + 49x_5 + 88x_6 + 78x_7 \\
&\text{s.t.} \\
&\qquad 98x_1 + 74x_2 + 94x_3 + 91x_4 + 51x_5 + 57x_6 + 57x_7 \leqslant 261, \\
&\qquad x_j \in \{0, 1\}, \quad j = 1, \ldots, 7.
\end{aligned}$$

Fig. 2 presents the efficiency of the items as calculated with the ratio (5), showing the points where the efficiency values are equal. The vertical lines separate the sub-regions in which the order of the items has changed (the bold line

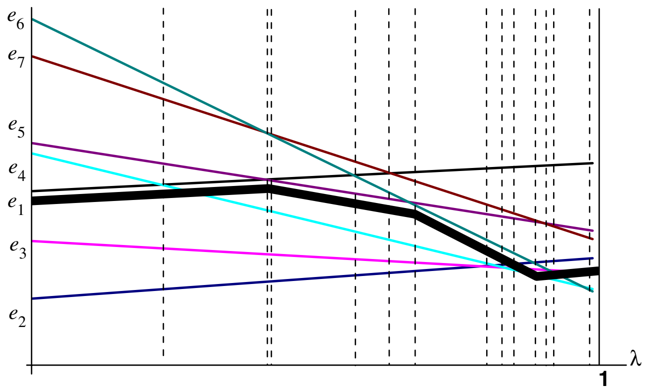

Fig. 2. Efficiency functions of the items.

Table 1
Size of the cores for efficient solutions in the different orders of the items

$\lvert C^{k,t}\rvert$	O_1	O_2	O_3	O_4	O_5	O_6	O_7	O_8	O_9	O_{10}	O_{11}	O_{12}	O_{13}	O_{14}
x^1	0	2	3	3	4	5	5	5	6	7	7	7	7	7
x^2	5	5	5	4	3	3	3	3	4	5	5	6	6	5
x^3	6	6	6	5	4	4	4	5	5	4	4	5	4	4
x^4	3	3	4	4	5	6	6	7	7	6	6	6	6	7
x^5	4	4	4	5	5	4	4	4	5	6	6	6	6	6
x^6	3	2	0	0	0	0	2	2	2	2	3	4	5	6
x^7	5	4	4	3	2	2	0	0	0	0	0	0	0	0

refers to the break item of the Dantzig solution and will be further explained in Example 3). As shown for this instance, 14 possible orders can be defined. The sub-regions correspond to the following ranges of λ:

$$[0; 0.230379], [0.230379; 0.415288]; [0.415288; 0.416667]; [0.416667; 0.572514];$$

$$[0.572514; 0.638643]; [0.638643; 0.671021]; [0.671021; 0.799320]; [0.799320; 0.825548];$$

$$[0.825548; 0.843705]; [0.843705; 0.894032]; [0.894032; 0.910138]; [0.910138; 0.922328];$$

$$[0.922328; 0.982020]; [0.982020; 1].$$

When solving the bi-criteria problem with an exact method, seven efficient solutions were found: $x^1 = (0001111)$, $x^2 = (1001001)$, $x^3 = (1010001)$, $x^4 = (0010111)$, $x^5 = (1001010)$, $x^6 = (1000011)$, $x^7 = (1000101)$. Solutions x^1, x^2, and x^3 are supported efficient solutions while x^4, x^5, x^6 and x^7 are non-supported efficient solutions. In the criteria space, the images are $z^1 = (97, 298)$, $z^2 = (139, 233)$, $z^3 = (147, 197)$, $z^4 = (105, 262)$, $z^5 = (125, 243)$, $z^6 = (129, 238)$ and $z^7 = (142, 199)$.

The composition of each efficient solution is then compared with that of each order, O_j, $j = 1, \ldots, 14$, and the size of the corresponding core is calculated. The results of these calculations are presented in Table 1. For example, sorting the values of the variables of solution x^1 according to order O_4 produces a core with size 3.

According to Definition 1, the size of the bi-criteria cores of the efficient solutions are 0, 3, 4, 3, 4, 0, and 0, respectively, obtained with λ belonging to [0; 0.230379], [0.572514; 0.8255483], [0.572514; 0.7993203] ∪ [0.8437054; 0.9101383] ∪ [0.9223284; 1], [0; 0.4152883], [0; 0.416663] ∪ [0.6386434; 0.799320], [0.416667; 0.825548], and [0.671021; 1]. They correspond to the smallest computed cores (i.e., using other orders associated with other sub-intervals could not lead to smaller cores).

If for an efficient solution x^t, the order defined in (6) corresponds to a sequence of 1's followed by a sequence of 0's, x^t can be obtained using the greedy procedure proposed by Dantzig [1]. This procedure produces the minimum core, which has a cardinality of 0. The existence of this order is given by the value of an optimal solution of the following linear problem:

$$
\begin{aligned}
\text{Max} \quad & \alpha_1 - \alpha_2 \\
\text{s.t.} \quad & \\
& \alpha_1 \leqslant \frac{\lambda c_j^1 + (1-\lambda)c_j^2}{w_j}, \quad j \in N_1^t, \\
& \alpha_2 \geqslant \frac{\lambda c_j^1 + (1-\lambda)c_j^2}{w_j}, \quad j \in N_0^t, \\
& \lambda \leqslant 1, \\
& \alpha_1, \alpha_2, \lambda \geqslant 0,
\end{aligned}
\tag{9}
$$

where $N_1^t = \{j : x_j^t = 1\}$ and $N_0^t = \{j : x_j^t = 0\}$.

Let $\alpha_1^*, \alpha_2^*, \lambda^*$ be an optimal solution of problem (9). For this problem the following propositions hold.

Proposition 2. *If* $\alpha_1^* - \alpha_2^* \geqslant 0$ *then* $x^t = \lfloor \overline{x} \rfloor$ *with* $\overline{x}$ *being an optimal solution of* $\max\{z(x, \lambda^*) : \sum_{j=1}^n w_j x_j \leqslant W, x_j \in [0, 1], j = 1, \ldots, n\}$, *where* $z(x, \lambda^*) = \lambda^* z_1(x) + (1-\lambda^*) z_2(x)$. *In this case, the cardinality of the corresponding core is* 0.

Proof. Let us suppose that $x^t \neq \lfloor \overline{x} \rfloor$, which means that either (a) $\exists j \in N_1^t : \lfloor \overline{x_j} \rfloor = 0$ or (b) $\exists j \in N_0^t : \lfloor \overline{x_j} \rfloor = 1$. Both cases lead to a contradiction with the fact of $\lfloor \overline{x} \rfloor$ being an optimal solution of the problem $\max\{z(x, \lambda^*) : \sum_{j=1}^n w_j x_j \leqslant W, x_j \in [0, 1], j = 1, \ldots, n\}$ since $e_j(\lambda^*) \geqslant e_l(\lambda^*), \forall j \in N_1^t, l \in N_0^t$ because $\alpha_1^* - \alpha_2^* \geqslant 0$. Given the definition of the core applied to x^t, the core size is 0. □

As shown in Example 1, the result presented in Proposition 2 can happen with either a supported or a non-supported efficient solution.

Corollary 1. *If* $\alpha_1^* - \alpha_2^* < 0$ *then a function of the type* $z(x, \lambda) = \lambda z_1(x) + (1-\lambda) z_2(x)$ *such that* $x^t = \lfloor \overline{x} \rfloor$ *cannot be defined. In this case, the core must have at least two items.*

Example 2. Consider the efficient solution $x^1 = (0001111)$ from Example 1. An optimal solution to problem (9) is $\alpha_1^* - \alpha_2^* = 0.17739409$ and $\lambda^* = 0$. Ordering the items according to their efficiency ratios using $z(x, \lambda^*)$, produces the items sequence: 6, 7, 5, 4, 1, 3, 2. This sequence corresponds to the variables values sequence: 1111000. Thus, x^1 can be obtained by applying the Dantzig [1] rule to the function $z(x, \lambda^*)$, and then rounding down the corresponding solution.

If $x^2 = (1001001)$ is considered, an optimal solution to problem (9) is $\alpha_1^* - \alpha_2^* = -0.2000774$ and $\lambda^* = 0.67102149$. Since $\alpha_1^* - \alpha_2^* < 0$, x^2 cannot be obtained by rounding down the solution using the Dantzig rule with any function $z(x, \lambda)$ since there is no λ that can define efficiency ratios for all variables with value 1 greater than those from the variables with a value of 0. Ordering the items according to their efficiency ratios using $z(x, \lambda^*)$ produces the items sequence 1, 7, 5, 6, 4, 3, 2. This sequence corresponds to the following variables values sequence: 1100100.

Bi-criteria {0, 1}-knapsack problems can have several Dantzig solutions. The number of such solutions is, however, bounded from above by the value mentioned in Proposition 1. The following proposition also holds true:

Proposition 3. *The number of Dantzig solutions for a bi-criteria* {0, 1}*-knapsack problem is equal to the number of extreme efficient solutions of the linear relaxation of that problem.*

The Dantzig solutions to a bi-criteria [0,1]-knapsack problem can be determined by using the bi-criteria simplex method with bounded variables, as explained by Gomes da Silva et al. [17]. A step-by-step explanation of the process of obtaining the Dantzig solutions is given below (Fig. 2 illustrates the results of this process):

Step 1: Consider the graphic representation of the efficiency ratio function (5) starting with $\lambda = 0$.

Step 2: Compute the efficiency ratios using the aggregate function $z(x, \lambda)$.

Step 3: Obtain the Dantzig solution and identify the break item, b. Graphically, this solution is kept the same until the point where the efficiency function of the break item crosses another function. At this point, the items with the upper efficiency functions are added to the knapsack until its full capacity is reached.

Step 4: Let λ^* be the smallest value, greater than λ, where at least one efficiency function crosses the break item function.

Step 5: If $\lambda^* \geqslant 1$, Stop; all the Dantzig solutions have been determined.

Step 6: Compute $B(\lambda^*) = \{l_j : e_{l_j}(\lambda^*) = e_b(\lambda^*)\}$ and remove all items pertaining to $B(\lambda^*)$ from the knapsack.

Step 7: Obtain a new Dantzig solution by filling the available knapsack capacity with the items pertaining to $B(\lambda^*)$, ordered according to their efficiency ratios, given $\lambda \leftarrow \lambda^* + \varepsilon$, where ε is a positive small perturbation. Identify the break item b, which can be the same as in Step 3.

Step 8: Go to Step 4.

Example 3. Applying the above procedure to the bi-criteria instance given in Example 1 produces the five Dantzig solutions shown in Fig. 2. These solutions are associated with the intervals of λ, respectively: [0; 0.230379], [0.23037; 0.415288], [0.415288, 0.671021], [0.671021, 0.894032], [0.894032, 1]. In Fig. 2, the bold horizontal line represents the efficiency of the break item of each Dantzig solution. These break items are 1, 1, 5, 6, and 2, respectively.

However, computing the bi-criteria core of an efficient solution requires more than comparing it with all the Dantzig solutions since the interval associated with the same Dantzig solution may be decomposed into sub-intervals whose item orders are different, and may be associated with a smaller core size. If only the orders corresponding to the Dantzig solutions are used, the results for the core size may be overestimated.

3. Numerical experiments related to the size of the bi-criteria core

This section reports on the computational experiments related to the size of the bi-criteria core. Five types of instances, in which $U(1, a)$ denotes a positive integer value not greater than a that has been generated randomly from an uniform distribution, were considered:

Type 1: $c_j^1, c_j^2, w_j \sim U(1, 100),\ j = 1, \ldots, n$ (uncorrelated instances, with small coefficients).

Type 2: $c_j^1, c_j^2, w_j \sim U(1, 10\,000),\ j = 1, \ldots, n$ (uncorrelated instances, with large coefficients).

Type 3: $c_j^1, c_j^2 \sim U(1, 100),\ w_j = 100,\ j = 1, \ldots, n$ (uncorrelated criteria functions, with small coefficients and constant weight);

Type 4: $c_j^1, w_j \sim U(1, 100), c_j^2 = w_j + 10,\ j = 1, \ldots, n$ (uncorrelated and strongly correlated criterion and weight-sum functions, with small coefficients);

Type 5: $c_j^1, w_j \sim U(1, 100), c_j^2 = 101 - c_j^1,\ j = 1, \ldots, n$ (uncorrelated criteria and weight functions and strongly correlated criteria functions, with small coefficients).

These instances differ in the way the coefficients were generated and in the range of the coefficients. They are inspired by the instance types considered by Martello and Toth [5] and Kellerer et al. [9]: uncorrelated instances with small and large coefficients and strongly correlated instances (Table 2).

Table 2
Characterization of efficient solutions: type 1 instances

n	# instances	$\overline{T}$	Type of solutions		Rounded Dantzig solutions	
			SS (%)	NSS (%)	DSS	DNSS
100	30	124.9	15.2	84.8	2.8	1.7
300	30	769.5	7.1	92.9	4	2.4
500	10	1754.6	4.9	95.1	4.3	4.1

Table 3
Characterization of efficient solutions: type 2 instances

n	# instances	$\overline{T}$	Type of solutions		Rounded Dantzig solutions	
			SS (%)	NSS (%)	DSS	DNSS
100	30	148.7	13.5	86.5	3.2	1.5
300	30	1100	4.8	95.2	3.4	3.6
500	10	2698.1	3.3	96.7	4.2	4.0

Table 4
Characterization of efficient solutions: type 3 instances

n	# instances	$\overline{T}$	Type of solutions		Rounded Dantzig solutions	
			SS (%)	NSS (%)	DSS	DNSS
100	30	326.5	10.6	89.4	34.0	0
300	30	2213.2	5.5	94.5	98.2	0
500	10	5894.4	3.3	96.7	189.3	0

Table 5
Characterization of efficient solutions: type 4 instances

n	# instances	$\overline{T}$	Type of solutions		Rounded Dantzig solutions	
			SS (%)	NSS (%)	DSS	DNSS
60	30	10.2	59.1	40.9	0.8	0.4
70	30	12.2	58.1	41.9	0.6	0.4
80	30	12	59.2	40.8	0.6	0.7

In all the instances, the knapsack capacity remains constant and is equal to 50% of the sum of the weights, which generally produces the highest number of efficient solutions [12,13].

The size of the bi-criteria core of exact efficient solutions to the bi-criteria {0, 1}-knapsack problem was evaluated in the following manner:

(1) the entire set of efficient solutions was generated using an implementation of the exact method proposed by Visée et al. [12];
(2) all the possible orders like (2) were generated;
(3) the bi-criteria core was computed for each efficient solution.

In the experiments, the number of variables changed according to the instance types because each type had a different level of difficulty. Types 4 and 5 instances, for example, are extremely difficult to solve using the branch-and-bound method by Visée et al. [12]. For this reason, only small instances were considered for these types of instances.

Tables 3–6 present the various sets of efficient solutions. In these tables, column 1 is the number of items; column 2 is the number of instances; column 3 is the average number of extreme efficient solutions ($\overline{T}$); columns 4 and 5 are the average percentage of supported solutions (SS) and non-supported solutions (NSS), respectively; and columns 6 and 7 are the average number of SS and NSS that are equal to the rounded Dantzig solutions (DSS and DNSS).

The average number of efficient solutions varies significantly, with instances types 4 and 5 instances being the extreme cases. The performance of types 1 and 2 instances are in the middle. For types 1–3 instances, the percentage of supported efficient solutions is considerably greater than the percentage of non-supported efficient solutions. This gap increases as the number of items increases. Types 4 and 5 instances have a balanced number of supported and

Table 6
Characterization of efficient solutions: type 5 instances

n	# instances	$\overline{T}$	Type of solutions		Rounded Dantzig solutions	
			SS (%)	NSS (%)	DSS	DNSS
40	15	3183.7	49.1	50.9	5.5	1.9
50	15	5102.2	35.3	64.7	6.1	2.1
60	10	16163.6	33.4	66.6	7.3	4.1

Table 7
Bi-criteria core results: type 1 instances

n	T	Type of solutions											
		Supported				Non-supported				Overall			
		$\frac{1}{2}T$	$\frac{3}{4}T$	$\overline{C}$	Range	$\frac{1}{2}T$	$\frac{3}{4}T$	$\overline{C}$	Range	$\frac{1}{2}T$	$\frac{3}{4}T$	$\overline{C}$	Range
100	3748	4	7	5.2	0–38	8	11	9.6	0–56	8	11	8.9	0–56
300	23 084	2	2.7	2.3	0–32	3.7	5	4.5	0–49.7	3.7	4.7	4.4	0–49.7
500	17 546	1.4	2	2.1	0–26.2	2.6	3.6	3.5	0–39.6	2.6	3.6	3.4	0–39.6

Table 8
Bi-criteria core results: type 2 instances

n	T	Type of solutions											
		Supported				Non-supported				Overall			
		$\frac{1}{2}T$	$\frac{3}{4}T$	$\overline{C}$	Range	$\frac{1}{2}T$	$\frac{3}{4}T$	$\overline{C}$	Range	$\frac{1}{2}T$	$\frac{3}{4}T$	$\overline{C}$	Range
100	4460	4	6	5.3	0–42	9	12	10.1	0–65	8	11	9.5	0–65
300	33 001	2	3	2.5	0–27.3	4	5.3	4.8	0–59.7	4	5	4.7	0–59.7
500	26 981	1.4	2	1.7	0–28.8	2.8	3.6	3.6	0–64.4	2.6	3.6	3.5	0–64.4

Table 9
Bi-criteria core results: type 3 instances

n	T	Type of solutions											
		Supported				Non-supported				Overall			
		$\frac{1}{2}T$	$\frac{3}{4}T$	$\overline{C}$	Range	$\frac{1}{2}T$	$\frac{3}{4}T$	$\overline{C}$	Range	$\frac{1}{2}T$	$\frac{3}{4}T$	$\overline{C}$	Range
100	9794	0	0	0	0–0	5	6	5.1	0–16	5	6	4.6	0–16
300	66 395	0	0	0	0–0	2	2.7	2.1	0–7	3	2.7	2.0	0–7
500	58 944	0	0	0	0–0	1.4	1.8	1.5	0–4.2	1.4	1.5	1.4	0–4.2

non-supported efficient solutions, with the average number of Dantzig solutions being very low, though slightly greater for SS than for NSS. Because of the mathematical characteristics of type 3 instances [18], all of the type 3 SS, and none of the non-supported ones, are Dantzig solutions. These instances also have the highest number of Dantzig solutions overall.

The results for the size of the bi-criteria core by instance type are presented in Tables 7–11. The table columns provide the total number of efficient solutions (T), and the percentage size of the core for the SS, NSS and for the entire set of efficient solutions. The columns $\frac{1}{2}T$, $\frac{3}{4}T$, $\overline{C}$, and range are, respectively, the maximum percentage core of

Table 10
Bi-criteria core results: type 4 instances

n	T	Type of solutions											
		Supported				Non-supported				Overall			
		$\frac{1}{2}T$	$\frac{3}{4}T$	$\overline{C}$	Range	$\frac{1}{2}T$	$\frac{3}{4}T$	$\overline{C}$	Range	$\frac{1}{2}T$	$\frac{3}{4}T$	$\overline{C}$	Range
60	299	8.3	11.2	8.1	0–35	11.2	15	11.5	0–30	10	13.3	9.5	0–35
70	366	7.1	10	7.3	0–21.4	10	14.3	10	0–30	8.6	11.4	8.5	0–30
80	371	7.5	8.8	6.7	0–16.3	7.5	11.3	8.1	0–26.3	7.5	8.8	7.3	0–26.3

Table 11
Bi-criteria core results: type 5 instances

n	T	Type of solutions											
		Supported				Non-supported				Overall			
		$\frac{1}{2}T$	$\frac{3}{4}T$	$\overline{C}$	Range	$\frac{1}{2}T$	$\frac{3}{4}T$	$\overline{C}$	Range	$\frac{1}{2}T$	$\frac{3}{4}T$	$\overline{C}$	Range
40	47 755	30	35	30.2	0–67.5	32.5	37.5	31.5	0–75	30	35	30.6	0–75
50	76 533	28	32	23.3	0–54	28	32	28	0–76	26	30	26.4	0–76
60	161 636	21.7	26.7	22.7	0–61.7	26.7	31.7	26.9	0–73.3	25	30	25.8	0–73.3

Table 12
Items changed in the bi-criteria core

n	$\|C\|$	$\|C^-\|$	$\|C^-\|/\|C\| \times 100$ (%)
100	56	2	3.57
300	149	40	26.85
500	198	7	3.54

50% of the total efficient solutions, the maximum percentage core of 75% of the total efficient solutions, the average percentage core, and the range of the percentage core.

The results show that, on average, the bi-criteria core is a very small percentage of the total number of items, with type 5 instances being an exception to this rule. The values in columns $\frac{1}{2}T$ and $\frac{3}{4}T$, respectively, indicate that 50% and 75% of solutions are found by exploring small neighborhoods around the break items of the weighted-sum functions. SS are easier to find in this exploration, as revealed by the smaller bi-criteria cores. This feature is a constant for all instance types. It is interesting to note that the average size of the bi-criteria core shrinks in relations to relative size as the overall size of the increases. The results obtained for these bi-criteria problems are very similar to those obtained for single-criterion problems. Inversely correlated instances with criteria functions (type 5 instances) appear to have the largest bi-criteria core size, while type 3 instances have the smallest, making the latter type the best instance on which to apply the core concept.

The apparent range of the bi-criteria core size is quite wide, with the maximum being considerably greater than the average size. However, detailed analysis reveals that, though the bi-criteria core is large, the number of variables in this core whose values are different than those in the corresponding continuous solution may be very small. This observation is clearly illustrated in the type 1 instances having the highest core size. Table 12 presents the data for the size of the core, $|C|$, as well as for the number of variables with a value different than the continuous solution, $|C^-|$, and the percentage of items that change with respect to the size of the bi-criteria core.

The most important conclusion that can be drawn from the above experiments concerns the consequences of the "compactness" of the bi-criteria core. This compactness is very interesting for the development of an approximate or an exact method for solving the {0, 1}-knapsack problem, since it allows the search to be reduced to a small set of

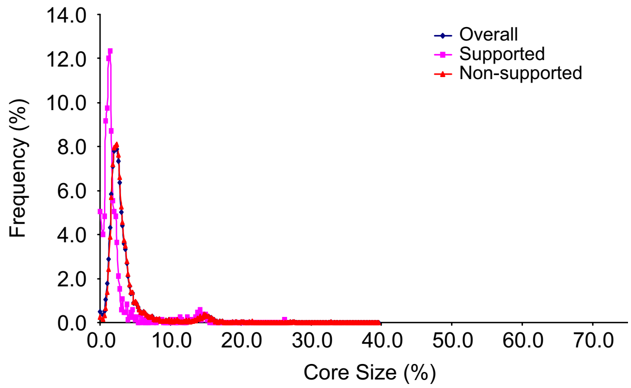

Fig. 3. Type 1 instances.

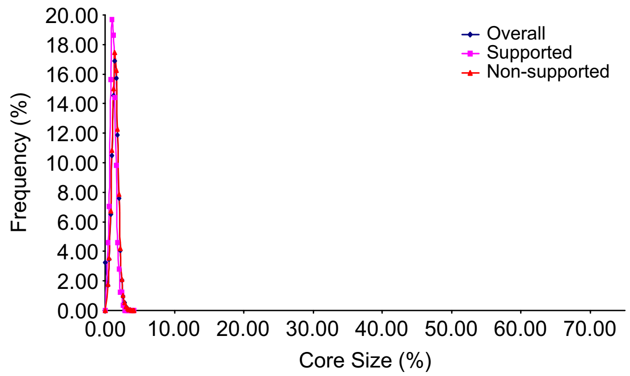

Fig. 4. Type 3 instances.

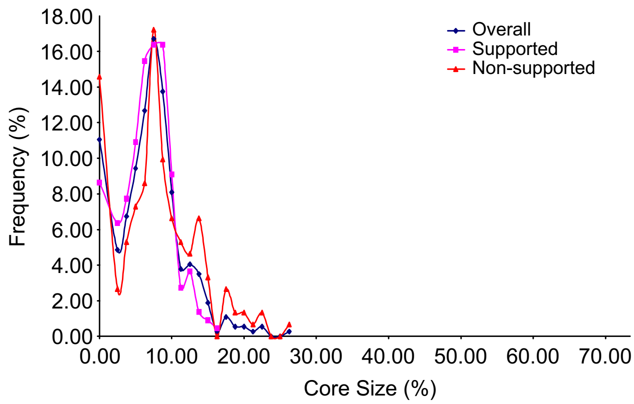

Fig. 5. Type 4 instances.

variables, revealing a preferred region in the decision space for a prioritary search for efficient solutions. Given the percentages shown in Table 12, a significant number of efficient solutions can be found in these small regions.

Figs. 3–6 show the distribution of the relative size of the bi-criteria core for the total efficient solutions and the supported, non-supported efficient solutions for types 1, 3–5 instances with the highest number of items. The same scale has been used for horizontal axes in these figures to facilitate the comparison of the distributions. As can be seen,

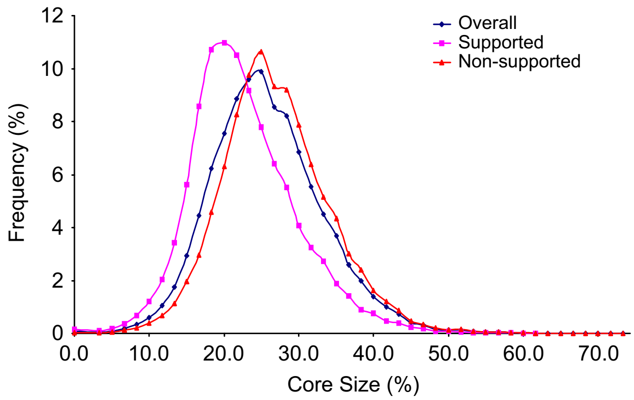

Fig. 6. Type 5 instances.

the distributions are biased, and very compact, especially for supported efficient solutions. The distribution pattern is similar for both SS and NSS.

4. Description of an approximate and an exact method

The experimental results reported above show that, using an appropriate weighted-sum function, a high percentage of efficient solutions is associated with small core size. Thus, it seems that reducing the search to a small subset of variables could produce a good approximate set of efficient solutions. For this reason, the search for efficient solutions should start by exploring the variables in the core. The main advantage of such a method is that the search is conducted from the beginning in a preferred region of the decision space, thus avoiding costly search operations.

An approximate method and an exact method for solving the bi-criteria {0, 1}-knapsack are described in the following sections.

4.1. An approximate method

The approximate method begins by computing the family of weighted-sum functions. Then, a core is defined for each of these functions (or a subset of functions), which is solved to obtain the potentially efficient solutions to the original problem. A filtering step is used to separate and save the non-dominated solutions in the solution set obtained. The algorithm proceeds as follows:

Step 1: Identify the family of weighted-sum functions, $\mathfrak{I}$.

Step 2: Consider a subset of $\mathfrak{I}$.

Step 3: Define a core for each weighted-sum function and define a bi-criteria core problem for each core (i.e., the problem whose variables pertain to each core).

Step 4: Solve each bi-criteria core problem exactly and update the set of efficient solutions (which was initially empty) with the obtained solutions.

Step 5: Consider the set of the efficient solutions as an approximate set of solutions that "solve" the bi-criteria {0, 1}-knapsack problem.

Step 4 of this algorithm can be executed with an existing exact method, a general branch-and-bound method, or dynamic programming based approach [19–22].

4.2. An exact method

Based on the experimental results presented in Section 3, the approximate set of efficient solutions appears to be very close to the exact one. Thus, an exact method could begin with the approximate set calculated above, and then, add another step to guarantee that the decision space is completely searched.

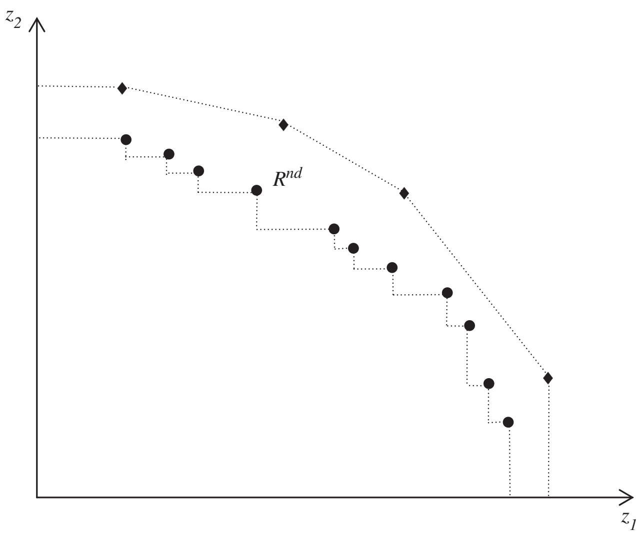

Fig. 7. Upper and lower frontiers of R^{nd}.

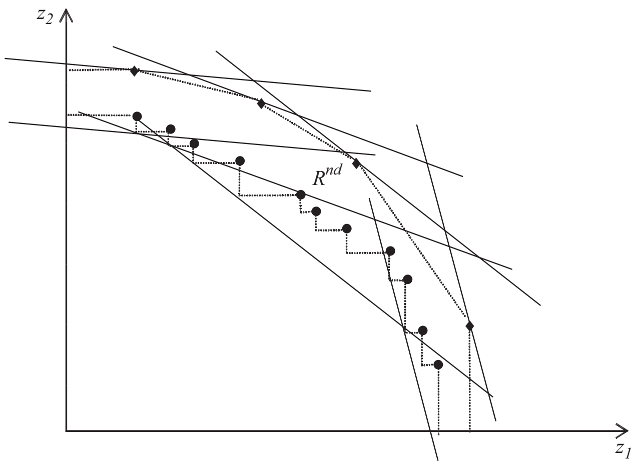

Fig. 8. Exploration of R^{nd} with different functions from $\Im$.

Step 1: Obtain an approximate set of efficient solutions, using the approximate method described above.

Step 2: Limit the search space in the criteria space. The upper bound of the criteria space is the line that connects the extreme non-dominated solutions from the linear relaxation (i.e., the images in the criteria space of the Dantzig solutions) and the lower bound is the stepped line that connects the solutions from the approximate set (region R^{nd} in Fig. 7).

Step 3: Define a branch-and-bound or dynamic programming method that completely explores the reduced criteria space.

Step 3 is crucial and not trivial. The weighted-sum functions $\Im$ could be used to define bands that cover the entire reduced criteria space (see Fig. 8). The upper and lower bounds for these functions (the limits of the bands) can be derived to set some variables to their "optimal" value in order to obtain non-dominated solutions within the bands. Once again, the reduced problems could be solved by using general approaches (or their adaptations) for solving multicriteria {0, 1} problems [19–22]. When all the bands have been completely explored, the result is the exact set of efficient solutions. This "banded" approach is described in detail in a previous paper by Gomes da Silva et al. [23].

5. Conclusions

In this paper, the concept of core was extended to the bi-criteria {0, 1}-knapsack domain. However, this extension is not trivial, since several cores can be defined for each efficient solution. The computational experiments conducted on five types of instances revealed that the characteristics of the single criterion case also hold true for the bi-criteria instances: they both have small size cores that increase slightly with the size of the problem. This parallel is due the hidden similarities that become apparent when solving problems (1) and (4). The results also showed that even in the worst cases of bi-criteria core size, very few variables of the continuous solution were changed. Based on these results, an approximate and an exact method for solving bi-criteria {0, 1}-knapsack problems were described briefly. The refinement and implementation of these methods will be tackled in future research.

Acknowledgements

The authors would like to acknowledge financial support of the MONET research project (POCTI/GES/37707/2001) and partially supported by the RAMS research project from CEG-IST.

References

[1] Dantzig G. Discrete variable extremum problems. Operations Research 1957;5:226–77.
[2] Balas E, Zemel E. An algorithm for large zero–one knapsack problems. Operational Research 1980;28:1130–54.
[3] Fayard D, Plateau G. An algorithm for the solution of the 0–1 knapsack problem. Computing 1982;28:269–87.
[4] Martello S, Toth P. A new algorithm for the 0–1 knapsack problems. Management Science 1988;34:633–45.
[5] Martello S, Toth P. Knapsack problems—algorithms and computer implementations. New York: Wiley; 1990.
[6] Pisinger D. An expanding-core algorithm for the exact 0–1 knapsack problem. European Journal of Operational Research 1995;87:175–87.
[7] Martello S, Pisinger D, Toth P. Dynamic programming and strong bounds for the {0, 1}-knapsack problems. Management Science 1999;45: 414–24.
[8] Pisinger D. A minimal algorithm for the {0, 1}-knapsack problem. Operations Research 1997;45:758–67.
[9] Kellerer H, Pferchy C, Pisinger C. Knapsack in problems. Berlin: Springer; 2004.
[10] Steuer R. Multiple criteria optimization, theory, computation and application. New York: Wiley; 1986.
[11] Ulungu EL, Teghem J. Solving multi-objective knapsack problem by branch-and-bound procedure. In: Clímaco J, editor. Multicriteria analysis. Berlin: Springer; 1997. p. 269–78.
[12] Visée M, Teghem J, Pirlot M, Ulungu EL. Two-phases method and branch and bound procedures to solve the bi-objective knapsack problem. Journal of Global Optimization 1998;12:139–55.
[13] Captivo M, Clímaco J, Figueira J, Martins E, Santos JL. Solving multiple criteria 0–1 knapsack problems using a labeling algorithm. Computers & Operations Research 2003;30:1865–86.
[14] Gandibleux X, Morita H, Katoh N. The supported solutions used as a genetic information in a population heuristic. In: Zitzler E, Deb K, Thiele L, Coello Coello CA, editors. Proceedings of the first international conference on evolutionary multi-criterion optimization. Lecture notes in computer science, vol. 1993. Berlin: Springer; 2001. p. 429–42.
[15] Gomes da Silva C, Figueira J, Clímaco J J. Integrating partial optimization with scatter search for solving bi-criteria {0, 1}-knapsack problems. European Journal of Operational Research 2007;177(3):1656–77.
[16] Gomes da Silva C, Clímaco J, Figueira J. A scatter search method for the bi-criteria knapsack problems. European Journal of Operational Research 2006;169(1):73–391.
[17] Gomes da Silva C, Figueira J, Clímaco J. An interactive procedure for the bi-criteria knapsack problem. Research Report, No. 4, INESC-Coimbra, Portugal, 2003, [in Portuguese] ⟨http://www.inescc.pt/download/RR2003_04.pdf⟩.
[18] Gomes da Silva C, Clímaco J, Figueira J. Geometrical configuration of Pareto frontier of bi-criteria {0, 1}-knapsack. Research Report, No. 8, INESC-Coimbra, Portugal, 2004.
[19] Bitran G. Linear multiple objective programs with zero–one variables. Mathematical Programming 1977;13:121–39.
[20] Bitran G. Theory and algorithms for linear multiple objective programs with zero–one variables. Mathematical Programming 1979;17(3): 362–89.
[21] Bitran G, Rivera J. A combined approach to solve binary multicriteria problems. Naval Research Logistics Quarterly 1982;29:181–201.
[22] Deckro R, Winkofsky E. Solving zero–one multiple objective programs through implicit enumeration. European Journal of Operational Research 1982;12:362–74.
[23] Gomes da Silva C, Figueira J, Clímaco J. An exact method for the bi-criteria {0, 1}-knapsack problem based on functions specialization. Research Report, No. 24, INESC-Coimbra, Portugal, 2004. ⟨http://www.inescc.pt/download/RR2004_24.pdf⟩.
[25] Pisinger D. Core problems in knapsack algorithms. Operations Research 1999;47(4):570–5.