Advances in Mathematical
Sciences and Applications
Vol. 29 , No. 2 (2020), pp. 345–363

GAKKOTOSHO
TOKYO JAPAN

THE PROBLEM OF THE COMPARTMENTALIZED KNAPSACK: A PROPOSAL OF THREE NEW HEURISTICS

Matheus Henrique Pimenta-Zanon *, Fabio Sakuray
Robinson Samuel Vieira Hoto

Universidade Estadual de Londrina - Brazil
Rodovia Celso Garcia Cid, PR-445, Km 380
Campus Universitário, Londrina - PR, 86057-970
(E-mail: matheus.pimenta@outlook.com, sakuray@uel.br, hoto@uel.br)

Abstract. The problem of the compartmentalized knapsack is classified as NP-Hard and has several models, both for linear and non-linear cases. The development of new heuristics for a resolution in a runtime useful for applications becomes necessary. In this work, three new heuristics are proposed, using the particularity of the linear model proposed by Inarejos (2017), which are considered only p_k compartments available for each class, being p_kX, $p_kGREEDY$ and $p_kMTComp$, the latter is based on the algorithm of Martello and Totti (1991). The heuristic $p_kMTComp$ presents solutions close to optimum value, being a promising method in solving the problem of the compartmentalized knapsack, when compared with other heuristics recognized in the literature.

*corresponding author: matheus.pimenta@outlook.com
Communicated by Editors; Received July 24, 2020.
AMS Subject Classification: 90-08, 90C05, 90C10, 90C59.
Keywords: Compartmentalized Knapsack Problem, Heuristic, Optimization.

1 Introduction

The problem of the compartmentalized knapsack consists in selecting a subset of items, in order to obtain the maximum utility, obeying the restrictions, as for example, the sum of the selected items dimension, must be inferior or equal the physical capacity of the knapsack [1, 2].

In this work will be considered only the compartmentalized knapsacks (CKP), a variation of the classical knapsack problem [3]. The CKP consists on the filling of limited capacity compartments, internal to a knapsack. The solution for the CKP maximizes the total utility of the knapsack [3, 4, 5, 6], considering the separation of the items in the internal compartments according to determined characteristics and limitations of the compartments/knapsack as for example the width or weigh of the items must meet the limits of the compartments/knapsack.

Several industrial processes that use one-dimensional cutting, such as coils or cutting bars (paper, clothes, plastics, steel, films, etc.), can be modeled as a compartmentalized knapsack (CKP). Resulting in the reduction of production costs, either by decreasing losses of raw material or by optimizing the production process [3, 5, 7, 8, 9, 10, 11, 12, 13, 14]. The steel coil cutting process, with two cutting steps, is illustrated in figure 1, with the following phases:

- First Phase: final items of the same characteristic (as an example of blade thickness) will be grouped into a category, or will be, are arranged in the same internal compartment as the knapsack. The categories and the unused material (loss or surplus) are demarcated and cut in this phase;

- Second Phase: processing execution, such as reducing the thickness of the blade. It can occur for categories and at different levels;

- Third Phase: marking the final items on the reel segment corresponding to its category, ocurring the cut in the final items.

In the figure 1 is observed that categories go through different phases, that is, category B needs processing prior to phase 3, while category A does not require such processing. Other examples of the application of the CKP can be accessed at [3, 15, 16].

Some of the heuristics presented in the literature proposed by [15] and [17] use solutions of the black box type to solve the subproblems generated in the development of the heuristics. Using this type of solution, it is not possible to access the methodology used for the determination of the solution. The p_k*Strong Capacities* heuristic presented in [15] is from the black box type, its goal is to create a variable number of compartments for each class, however, its use is limited to problems of reduced dimensions, that is, dozens of classes and items.

Other heuristics that do not use black box solutions in their development, such as *Z-best compartments* and $W-$*capacities* presented in [6] are the main works of literature for CKP resolution. In the heuristic *Z-best compartments*, the Z compartments with the best utility associated to each class of items are considered, while in the heuristic $w-$*capacity*, for each class of items the w best capacities are calculated for each compartment.

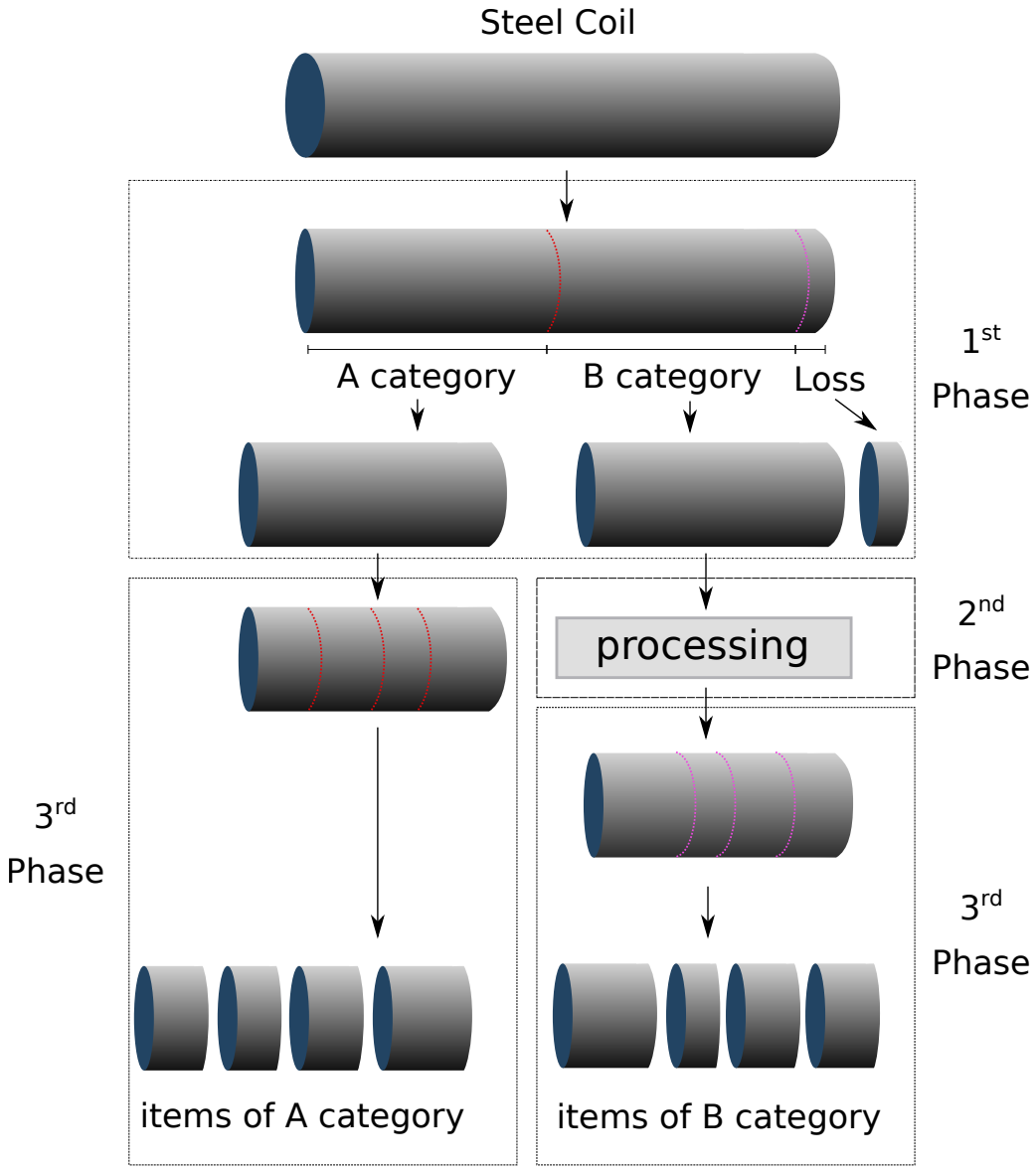

Figure 1: Steel Coil Cutting Process.

The Compartmented Knapsack Problem has non-linear approaches [6, 10, 12, 18, 19] and linear [4, 15, 17], we present below some of them.

Cruz in [17] suggests a linear heuristic approach to the CKP nonlinear model. Linear modeling presents results close to the optimum value with reduced execution time. Arenales et. al presents in [19] a new linear model for the same problem based on nonlinear systems, the heuristics offered as a solution are evaluated by the difference with the optimal value for the problem, obtained through proprietary software, the heuristic has limitations in relation to the number of items to be processed, dozens of items per class are simulated only.

Quiroga-Orozco in [15] indicates a new model, defined as a Strong Linear Model for the Compartmented Knapsack Problem along with a new heuristic, using this new model.

Marques and Arenales in [6] present the heuristic called $w - capacity$ for the non-linear case of the CKP, applying to steel coil cutting problems with more than one cutting step. The heuristic proposed by [15] provides an approximation similar to the heuristic *w-capacities*, however with a runtime superior to the heuristic *w-capacities*.

The applications of the Compartmented Knapsack Problem are mostly for cutting and packaging problems [4, 6, 10, 12, 15, 18, 19], which motivated the application of the heuristics proposed in this work.

The proof of the linearity of the Compartmented Knapsack Problem is performed by [4] being the model used in this article, which uses exhaustive and decomposition methods in its construction. Exhaustive methods have a long execution time due to the realization of all possible combinations to determine the best solution of the problem, whereas in the decomposition methods an initial problem is decomposed into subproblems to obtain the final solution.

The objective of this work is to present heuristics for CKP solution, with processing time and solution precision suitable for commercial use, even if a volume of items for processing in the order of tens of thousands is used. To evaluate the results obtained, results from the heuristics *w - capacities* and the value considered to be optimal (linear model presented by [4]) will be used.

In addition to this introduction, this article is organized as the following: section II will present the proposed heuristics, in section III the numerical simulations will be presented and a discussion about the simulations and in section IV the conclusions will be presented.

2 Proposed Heuristics

The proposed heuristics present a solution to the unbounded optimization problem, described by [4]:

$$\text{Maximize: } z = \sum_{k=1}^{q} \sum_{i \in N_k} \sum_{j=1}^{p_k} u_i a_{ij}^k \tag{1}$$

$$\text{Subject to: } \sum_{k=1}^{q} \sum_{i \in N_k} \sum_{j=1}^{p_k} l_i a_{ij}^k \leq L \tag{2}$$

$$\delta_j^k L_{min}^k \leq \sum_{i \in N_k} l_i a_{ij}^k \leq \delta_j^k L_{max}^k \quad \text{with } j = 1, \ldots, p_k \quad k = 1, 2, \ldots q \tag{3}$$

$$\sum_{k=1}^{q} \sum_{j=1}^{p_k} \delta_j^k \leq F_1 \tag{4}$$

$$\sum_{i \in N_k} a_{ij}^k \leq F_2 \ , j = 1, 2, \ldots, p_k \ k = 1, 2, \ldots, q \tag{5}$$

$$\delta_j^k \in \{0, 1\} \text{ e } a_{ij}^k \geq 0 \text{ and integers, } \ i \in N_k, \ j = 1, 2, \ldots, p_k, \ k = 1, 2, \ldots, q \tag{6}$$

The objective function 1 seeks to maximize the total utility (u_i) of the items, the constraint 2 indicates the physical capacity of the L knapsack, with l_i representing the physical characteristic of the i item. The 3 constraint provides the lower and upper limits for the compartments of each class, only non-null compartments will be included, that is $\delta_j^k = 1$. The restrictions 4 and 5 are knife 01 and knife 02 of the cutting process, respectively, which inform the number of non-null compartments that will be inserted in the knapsack and the number of items that will be inserted in each compartment , respectively, finally the constraint 6 presents the domain of the variables.

For the elaboration of the proposed heuristics, it is strongly used the fact that the number of compartments to be created in the linear model cannot exceed $\sum_{k=1}^{q} p_k$. In addition to the use of the Inarejos [4] linear model, the development of viable compartments based on Hoto [3, 20] is also used. The value of p_k is chosen for the selection of viable compartments.

The three heuristics proposed share the same initial stage, in which all viable compartments are defined and an initial utility is also associated for each compartment. A compartment is defined as viable if there is a non-negative integer linear combination of the weights of the associated class items, equal to the capacity of the compartment concerned.

Property 1: Consider compartment j of class k of capacity $w_j > l_{min}^k$ and the weights associated with that class N_k. If there is any item $r \in N_k$ such that the capacity compartment $w_j - l_r$ is viable, then the capacity compartment w_j is viable.

Proof. On the hypothesis that $w_j - l_r$ is viable, it follows that $w_j - l_r = \sum_{i \in N_k} l_i a_i$, of this one has $w_j = l_r(a_r + 1) + \sum_{\substack{i \in N_k \\ i \neq r}} l_i i a_i$. □

After generating viable compartments, they are ordered in decreasing order of efficiency $\frac{U_j}{W_j} > \frac{U_{j+1}}{W_{j+1}}$ and are selected the most efficient p_k compartments. The initial process algorithm is given in Algorithm 1.

After generating all viable compartments, represented by the j index, a restricted knapsack problem is formulated to obtain the best utilities associated with each built compartment. The way to obtain these optimum values of the utilities is obtained by solving the problem (7) - (10):

$$\text{Maximize } U_j = \sum_{i \in N_k} u_i a_{ij} \tag{7}$$

$$\text{Subject to: } \sum_{i \in N_k} l_i a_{ij} \leq w_j \tag{8}$$

$$\sum_{i \in N_k} a_{ij} \leq F_2 \tag{9}$$

$$a_{ij} \geq 0 \text{ and integer}, i \in N_k \tag{10}$$

Algorithm 1 Obtaining the Viable Compartments and Initial Utilities

Require: l_i, u_i, l_{max} e l_{min}
Ensure: W e U
Be CL^k set of all available class widths k.
Initialization: For each k class do:
Sort the item widths in ascending order.
while $w_j \leq l_{max}$ **do**
  if $w_j \in CL$ **then**
    $w_j \in C$
    $ut_j = u_j$
    $w_j = w_j + 1$
  end if
  if $w_j - l_i \in CL$ **then**
    $C = C \cup w_j$
    $i = i + 1$
    $ut_j = u_j + ut_i$
    $w_j = w_j + 1$
  end if
end while
for all $w_j \in [l_{min}, l_{max}]$ **do**
  $W_k = \cup w_j$
  $U_k = \cup ut_j$
end for

Right after obtaining the best utilities associated with each of the p_k viable compartments of each class k, the data U_j and W_j will be the new values for solving the entire programming problem given below, which the viable solution for the CKP will result. W_k are the indexes associated with each viable compartment.

$$\text{Maximize Z} = \sum_{i \in W_k} U_i y_{ij} \tag{11}$$

$$\text{Subject to:} \sum_{i \in W_k} W_i y_{ij} \leq L \tag{12}$$

$$\sum_{i \in W_k} y_{ij} \leq F_1 \tag{13}$$

$$y_{ij} \geq 0 \text{ and integer}, i \in W_k \tag{14}$$

The next subsections present the details of the heuristics proposed in this article, named: p_kX, $p_kGREEDY$ and $p_kMTComp$.

2.1 Heuristic p_kX

To obtain a viable solution for CKP, the heuristic denominated as p_kX uses the *software* FICO® Xpress [21] with its FICO optimization package FICO® Xpress *Optimizer*.

The optimization package is of the proprietary type, that is, owned by the developer of *software*, it is not possible to detail the internal procedures adopted by the *maximize* function of the optimization package. The p_kX heuristic uses the *maximize* function to solve the subproblems: (7) - (10) and (11) - (14). The heuristic is presented by the Algorithm 2.

Algorithm 2 Heuristic p_kX

Require: l_i, W_j, U_j, u_i, l_{max}, l_{min}, L, q, p_k e n
Ensure: x_j e Z
1: **for all** k em $1, \ldots, q$ **do**
2: Sort the item widths in ascending order.
3: Create the viable compartments by executing the algorithm 1.
4: Determine the efficiency of each viable compartment.
5: Select the most efficient p_k compartments.
6: Solve the problem (7) - (10) using the function *maximize*.
7: **end for**
8: Solve the problem (11) - (14) using the function *maximize*.

2.2 Heuristic $p_kGREEDY$

In search for a shorter execution time, the heuristic called $p_kGREEDY$ is developed, which does not use a proprietary package of *software* to solve the CKP.

For the resolution of the CKP, the heuristic $p_kGREEDY$, uses the greedy method which consists of inserting the largest number of items of greater efficiency inside the knapsack, if it is still possible to insert more items, the next item of bigger efficiency is arranged in its interior, and so on.

Mathematically, the value of $x_1 = \left\lfloor \frac{L}{l_1} \right\rfloor$ and from x_2 are inserted $\left\lfloor \frac{L - \sum_{i=1}^{j-1} l_i x_i}{l_j} \right\rfloor$ items for each x_j. This way, it is performed only once, forming only one branch of the enumeration tree.

The 3 algorithm details the $p_kGREEDY$ heuristic.

Algorithm 3 Heurística $p_kGREEDY$

Require: l_i, W_j, U_j, u_i, l_{max}, l_{min}, L, q, p_k e n
Ensure: x_j e Z
1: **for all** k em $1, \ldots, q$ **do**
2: Sort the item widths in ascending order.
3: Create the viable compartments by executing the algorithm 1.
4: Determine the efficiency of each viable compartment.
5: Select the most efficient p_k compartments.
6: Solve the problem (7) - (10) using the greedy method.
7: **end for**
8: Solve the problem (11) - (14) using the greedy method.

2.3 Heuristic $p_kMTComp$

The $p_kMTComp$ heuristic is based on the exact resolution algorithm of Martello and Toth [22]. This algorithm is modified to include the knives restriction of the linear model proposed by Inarejos [4].

Two algorithms from Martello and Toth [1] were used, the first is a method of *branch and bound* and the second uses what is called a core problem.

The first Martello and Toth [1] algorithm that was used is called MTU1, which consists of a *branch and bound* method where the implicit enumeration tree has $n+1$ search levels. For the construction of the enumeration tree, the items must be in decreasing order of efficiency. As it is a *branch and bound* method in each visited node, an analysis is carried out in relation to the best solution, if lower, this node is closed and the beginning of a new branch is returned. The details of the algorithm together with its explicit visualization can be accessed at [1].

The modification of the method, for the inclusion of the knives, occurs at the stage when the items are inserted inside the knapsack, in the original algorithm $y = \left\lfloor \frac{\hat{L}}{l_j} \right\rfloor$ items,

with the adaptation, its inserted $y = min\left\{\left\lfloor\frac{\hat{L}}{l_j}\right\rfloor, F\right\}$, where F represents the knife in use at the time the method is executed.

For the implementation of the MTU1 algorithm, the C programming language was used, using the finite state machine model, in order to eliminate the so-called *goto* of the original code. It was also possible to use *multithreading* in the implementation of the $p_kMTComp$ heuristic, for each CKP class a *thread* is executed in a processor core. The MTU1 method does not solve problems in which the number of items available is greater than 250000 items, with the objective of eliminating this limitation, the MTU2 algorithm will be used.

The second algorithm of Martello and Toth [1] to be used is called MTU2, it is used for the resolution of large specimens, such as 250000 items available for the selection of the knapsack. The MTU2 algorithm uses MTU1 internally to solve the core problem, which uses only the most efficient items.

The detailed algorithm and examples can be accessed at [1]. Following is the 4 algorithm, which details the $p_kMTComp$ heuristic.

Algorithm 4 Heuristic $p_kMTComp$

Require: l_i, W_j, U_j, u_i, l_{max}, l_{min}, L, q, p_k e n
Ensure: x_j e Z
1: **for all** k em $1, \ldots, q$ **do**
2: Sort the item widths in ascending order.
3: Create the viable compartments by executing the algorithm 1.
4: Determine the efficiency of each viable compartment.
5: Select the most efficient p_k compartments.
6: Solve the problem (7) - (10) using the MTU2 algorithm.
7: **end for**
8: Solve the problem (11) - (14) using the MTU2 algorithm.

The next section will present the numerical simulations and analysis of the heuristics.

3 Simulations and Discussion

For the simulations of the linear model and the p_kX heuristic, the *software* FICO® Xpress [21] for 64-bit architecture was used.

For the implementation, execution of the heuristics $p_kGREEDY$ and $p_kMTComp$ and for the ordering of items and classes by efficiency, the C programming language was used. The *hardware* used in all computational simulations and experiments consist of an Intel processor ® Inside™ Xeon® CPU W3520, 8 GB of RAM, Microsoft Windows operating system ® Server 2012 R2 Standard, which is in the simulation laboratory of the Mathematics Department of the State University of Londrina (UEL).

The data to perform the simulations were organized into five classes with sizes defined in $q = 5, 10, 20, 50$ and 100; with five subcategories per class, representing the number of

items available in each class, being $n = 10, 50, 100, 1000$ and $10,000$. Table 1 shows the organization of classes and items.

Table 1: Subcategories (*Sets*) for Simulations

		Categories (q)				
		5	10	20	50	100
Items by Category (n)	10	5/10	10/10	20/10	50/10	100/10
	50	5/50	10/50	20/50	50/50	100/50
	100	5/100	10/100	20/100	50/100	100/100
	1000	5/1000	10/1000	20/1000	50/1000	100/1000
	10000	5/10000	10/10000	20/10000	50/10000	100/10000

For the creation of the specimens, data were used based on real problems of cutting steel coils in two stages [3], with the following elements: total capacity of the knapsack $L = 1100mm$, capacity of the compartments of each class are limited between the values L_{min}=$154mm$ and L_{max}=$456mm$, item widths will be generated with values evenly distributed between $53mm$ and $230mm$, utility items will be values between 1 and 100. For each q/n category, 100 copies were produced using a random generator, based on Gau and Washer [23], resulting in 2500 copies.

Each of the 2500 copies were ordered through the efficiency of its items. After the initial organization of the copies, simulations were performed for the linear model and the heuristics p_kX, $p_kGREEDY$ and $p_kMTComp$, after this were elaborated the Tables 3, 4 and 5 where the results obtained are detailed, the maximum execution time for each copy was fixed at 86400 seconds.

All simulations took place on the same equipment, within the same specifications, with no changes occurring during the execution of all copies, the equipment was dedicated to the execution of the simulations.

The representation through letters occurs as follows the category each row of the tables 2, 3, 4 and 5 represent one of the subcategories presented in the table 1, that is, A represents the results of the processing of the subcategory 5/10, the B refers to 5/50 and so on. The following information is presented in these tables: average execution time in seconds ($\overline{T}$), standard deviation of the execution time in seconds ($\sigma(T)$) and the percentage refers to the difference between the heuristic linear (z_{linear}) and the heuristic under analysis ($z_{heuristic}$) in the respective table (gap). The gap is given by:

$$gap = \frac{z_{linear} - z_{heuristic}}{z_{linear}}$$

In the tables, the average values are analyzed ($\overline{gap}$), minimum (gap_{min}) and maximum (gap_{max}), obtained among the samples performed in the heuristic analysis.

Table 2 presents the results obtained by solving the specimens using the linear model. Table 3 presents the statistics related to the p_kX heuristic compared to the linear model. For copies of classes 50/10000 (T), 100/1000 (X) and 100/10000 (Y), the execution time is longer than the maximum time of 86400 seconds, these results are being represented by $*$. Table 4 details the results obtained through the $p_kGREEDY$ heuristic and Table 5

presents the statistics related to the $p_kMTComp$ heuristic compared to the linear model proposed by Inarejos [4].

Table 2: Linear Model Statistics

Set	Subcategory	*Linear*	
		$\overline{T}$	$\sigma(T)$
A	5/10	217,6535	658,0430
B	5/50	43,7511	420,2747
C	5/100	0,0601	0,0321
D	5/1000	0,1930	0,0443
E	5/10000	3,2295	0,1215
F	10/10	1,5687	9,5373
G	10/50	1,5936	15,3062
H	10/100	0,1157	0,0493
I	10/1000	0,4233	0,0464
J	10/10000	12,1816	0,2539
K	20/10	29,9757	181,3964
L	20/50	0,1521	0,2808
M	20/100	18,9978	187,6012
N	20/1000	1,0560	0,0962
O	20/10000	100,902	0,8131
P	50/10	0,1070	0,2117
Q	50/50	0,4731	1,1416
R	50/100	26,4013	257,8968
S	50/1000	4,8865	0,3422
T	50/10000	1058,958	14,3281
U	100/10	23,7795	235,5939
V	100/50	26,6162	257,8439
X	100/100	26,6663	252,8972
W	100/1000	30,9031	1,3370
Y	100/10000	5226,72	64,4448

The pictures 2 and 3 present the behavior of the heuristics related to the $\overline{gap}$ and to the $\overline{T}$.

In order to evaluate the behavior of the $p_kMTComp$ heuristic in relation to other heuristics recognized in the literature, comparisons were made with the $w - capacities$ heuristic, proposed by Marques [6].

The $w - capacities$ heuristic was developed for the unbounded case of the compartmentalized knapsack, Table 6 presents the results obtained through simulations performed with the $w - capacity$ heuristic. For the categories with $n = 1000$ and $n = 10000$ the $w - capacities$ heuristic did not obtain solutions, this is due to its implementation with the method of Gilmore and Gomory [24] for the resolution of the internal subproblems of heuristic, which solves problems with up to hundreds of items. These categories are represented by $*$ in Table 6.

Table 3: Heuristic p_kX Statistics

Sets	p_kX					
	$\overline{T}$	$\sigma(T)$	$\overline{gap}$	$\sigma(gap)$	gap_{min}	gap_{max}
A	1,065	0,042	0,03%	0,24%	0%	2,23%
B	2,691	0,096	0,51%	1,26%	0%	5%
C	3,036	0,455	3,04%	2,98%	0%	11,10%
D	13,172	0,167	0,78%	1,25%	0%	3,99%
E	8276,237	32,381	0,01%	0,02%	0%	0,11%
F	2,679	0,455	0,11%	0,26%	0%	0,98%
G	12,586	0,654	0,84%	1,00%	0%	5,56%
H	13,491	0,235	1,98%	2,11%	0%	10,53%
I	40,092	0,367	0,64%	1,18%	0%	3,20%
J	16624,740	59,062	0,01%	0,01%	0%	0,05%
K	14,949	1,953	0,04%	0,36%	0%	3,57%
L	71,314	1,649	0,36%	1,04%	0%	3,92%
M	76,112	0,739	1,19%	1,57%	0%	6,68%
N	185,208	0,969	0,46%	1,09%	0%	3,13%
O	34152,200	241,510	0%	0%	0%	0%
P	202,236	40,218	0,02%	0,08%	0%	0,38%
Q	900,150	9,321	0,19%	0,59%	0%	2,31%
R	961,270	6,144	0,35%	0,69%	0%	3,25%
S	2420,670	32,676	0,01%	0,06%	0%	0,57%
T	*	*	*	*	*	*
U	1560,647	267,478	0,10%	0,74%	0%	6,18%
V	6698,394	53,293	0,06%	0,37%	0%	2,22%
W	7216,887	32,286	0,14%	0,44%	0%	2,84%
X	*	*	*	*	*	*
Y	*	*	*	*	*	*

The pictures 4 and 5 present the heuristics behavior $p_kMTComp$ and $w - capacity$ related to the $\overline{gap}$ and to the $\overline{T}$.

3.1 Simulation analysis

The execution of the specimens of each category provided results that allow to analyze in detail the behavior of the three proposed heuristics.

The p_kX heuristic proved to be, among the three proposed heuristics, the one with the longest execution time, in some cases presenting average times greater than one day of execution for each copy (Table 3 shows this). Regarding the quality of the solutions obtained, 100% of the solutions in the categories obtained were less than 5% of the optimum value, showing reliable solutions in relation to the optimum. Another highlight is the reduced standard deviation of the solutions, showing that the discrepancy between the values is low. Regarding the amplitude of the *gap* intervals obtained, in 91.3% of

Table 4: Heuristic $p_kGREEDY$ Statistics.

Sets	$p_kGREEDY$					
	$\overline{T}$	$\sigma(T)$	$\overline{gap}$	$\sigma(gap)$	gap_{min}	gap_{max}
A	0,0009	0,0035	14,33%	4,34%	1,11%	26,53%
B	0,0156	0,0049	10,64%	4,31%	0%	20,33%
C	0,0241	0,0079	10,77%	6,50%	1,14%	30,59%
D	0,1011	0,0080	14,85%	9,15%	0,51%	30,93%
E	8,4886	0,0154	15,03%	7,47%	0,63%	31,15%
F	0,0036	0,0064	9,74%	5,98%	1,17%	20%
G	0,0314	0,0081	11,13%	6,05%	0%	31,53%
H	0,0506	0,0082	10,64%	6,94%	1,14%	34,33%
I	0,2023	0,0058	12,92%	8,98%	0,51%	30,51%
J	16,9356	0,0174	15,41%	6,71%	1,38%	31,18%
K	0,0058	0,0073	8,30%	8,71%	0%	27,27%
L	0,0631	0,0099	10,10%	5,26%	0%	37,86%
M	0,1021	0,0126	8,63%	6,53%	0%	24,36%
N	0,4021	0,0081	12,97%	8,55%	0%	30,84%
O	33,9172	0,0235	14,51%	8,89%	1,69%	31,22%
P	0,0150	0,0044	8,06%	8,30%	0%	25,96%
Q	0,1563	0,0162	10,75%	6,61%	0%	34,69%
R	0,2528	0,0215	9,50%	6,25%	0%	30%
S	1,0054	0,0086	14,89%	7,26%	0,40%	31,69%
T	84,6561	0,0599	14,52%	9,83%	1,74%	31,22%
U	0,0297	0,0053	9,11%	7,45%	0%	31,37%
V	0,3148	0,0269	9,58%	5,60%	0%	30%
W	0,4994	0,0298	7,77%	6,54%	0%	25%
X	2,0107	0,0079	16,14%	8,24%	0,41%	32,18%
Y	169,2361	0,0922	13,73%	9,93%	1,74%	31,22%

the tests performed, this interval was less than 10%, denoting small intervals. Although the execution time indicates high values (see Figure 3), the p_kX heuristic demonstrates quality in the solutions obtained (see Figure 2).

The $p_kGREEDY$ heuristic fills the knapsack with the most efficient items, this feature allows less execution time compared to p_kX heuristics. On the other hand, due to this characteristic, the solutions present *gap* higher than the heuristics in comparison, p_kX and $p_kMTComp$. From the results obtained through the experiments performed, only 32% has an average *gap* less than 10%, and none of the categories presented an average *gap* less than 5%.

Another situation unfavorable to this heuristic is the amplitudes of the *gap* intervals with intervals greater than 20% in all simulated categories, another factor to be noted is that in 52% of the simulated categories, in none of the 100 examples of each category the optimum was found as a solution, presenting the worst performance in relation to the solutions obtained in comparison with the heuristics p_kX and $p_kMTComp$.

Table 5: Heuristic $p_kMTComp$ Statistics.

Sets	$p_kMTComp$					
	$\overline{T}$	$\sigma(T)$	$\overline{gap}$	$\sigma(gap)$	gap_{min}	gap_{max}
A	0,0018	0,0048	1,21%	1,78%	0%	7,14%
B	0,0108	0,0285	1,64%	2,55%	0%	11,42%
C	0,0112	0,0065	2,03%	1,73%	0%	5,82%
D	0,0575	0,0102	1,12%	1,54%	0%	6,12%
E	1,8970	0,2344	1,10%	1,42%	0%	3,84%
F	0,0045	0,0069	1,14%	1,48%	0%	10,20%
G	0,0115	0,0068	1,58%	1,85%	0%	5,56%
H	0,0063	0,0082	1,58%	1,54%	0%	6,15%
I	0,0909	0,0112	0,99%	1,25%	0%	3,61%
J	3,4362	0,2815	0,35%	0,63%	0%	3,54%
K	0,0070	0,0075	0,66%	1,77%	0%	6,32%
L	0,0214	0,0070	0,60%	1,24%	0%	5,26%
M	0,0350	0,0070	1,28%	1,45%	0%	6,68%
N	0,1624	0,0145	0,99%	1,21%	0%	3,55%
O	6,7815	0,3451	0,18%	0,09%	0,05%	0,48%
P	0,0111	0,0290	0,22%	0,73%	0%	4,21%
Q	0,0487	0,0080	0,41%	0,89%	0%	5,26%
R	0,0792	0,0120	0,66%	1,06%	0%	5,48%
S	0,4030	0,0198	0,96%	1,62%	0%	7,39%
T	17,3080	1,6290	0,11%	0,06%	0,05%	0,26%
U	0,0270	0,0380	0,64%	1,50%	0%	6,90%
V	0,1028	0,0114	0,33%	0,83%	0%	5,26%
W	0,1582	0,0224	0,63%	0,98%	0%	3,28%
X	0,7968	0,0251	0,96%	1,25%	0%	6,09%
Y	35,0490	2,5510	0,09%	0,05%	0%	0,21%

Finally, the $p_kMTComp$ heuristic among the three proposed heuristics showed results that indicate the best performance for the considered specimens. The fact that this heuristic does not depend on proprietary software with embedded solutions to solve the heuristic subproblems is a differential when compared to the p_kX heuristic. The use of the Martello and Toth cite MARTELLO1990b method enabled solutions with low *gap* values, 100% of the average values obtained from the categories were less than 5%, with 96% of the average values obtained were less than 2% of the optimal solution and the standard deviation shows that 96% of the values are below 2% (Table 5 shows this).

Regarding the *gap* intervals obtained, 92% were below 10%, indicating small variations in the results obtained. Another factor to be noted is for categories with $n = 10000$, which have better performance indicators for the solutions obtained.

Regarding the execution time, the results obtained in these specimens indicate a final execution time lower than the heuristic p_kX and in some categories values lower than the heuristic $p_kGREEDY$.

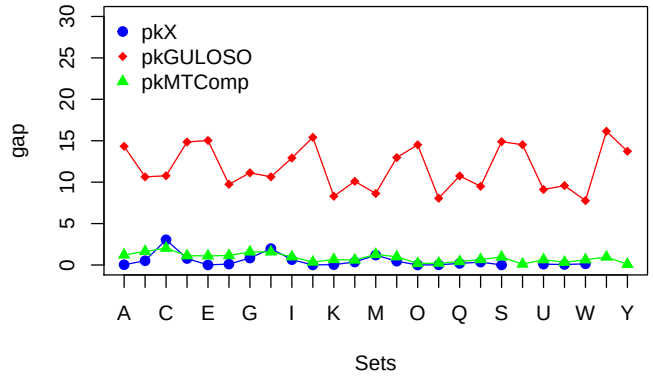

Figure 2: Heuristics performance - $\overline{gap}$.

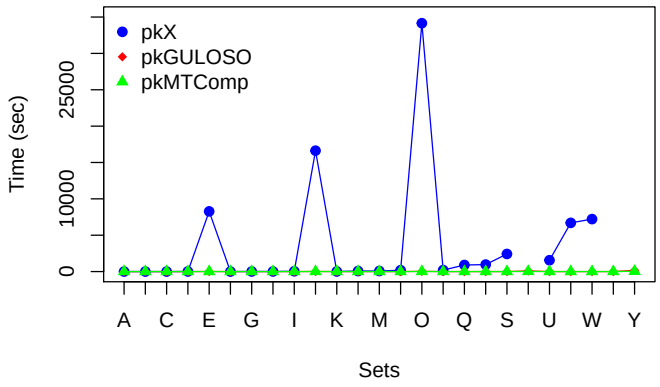

Figure 3: Heuristics performance - $\overline{T}$.

As shown by Tables 3, 4 and 5 the $p_kMTComp$ heuristic has a superior performance compared to the p_kX and $p_kGREEDY$ heuristics, with the results obtained indicating that it is a promising heuristic among the three developed.

When looking at the execution time of the three heuristics (see Figure 3), the indicators show that the $p_kMTComp$ heuristic presents a lower execution time than the p_kX and $p_kGREEDY$ heuristics.

Table 6 presents the results obtained through the experiments carried out with the heuristic $p_kMTComp$ and $w - capacities$ which indicate a superior performance of the heuristic $p_kMTComp$ in relation to the obtained solution, producing solutions with a lower gap compared to the $w - capacity$ heuristic. Another detail where the $p_kMTComp$ heuristic is superior to the $w - capacities$ heuristic is in solving examples with thousands of items, something not accomplished by the $w - capacities$ heuristic. The use of the Martello and Toth [22] method allows the execution of copies with thousands of items,

Table 6: Heuristica $w - capacities$ Statistics.

Sets	$w - capacidade$					
	$\overline{T}$	$\sigma(T)$	$\overline{gap}$	$\sigma(gap)$	gap_{min}	gap_{max}
A	0	0	5,32%	7,67%	0%	28,11%
B	0,0003	0,0020	16,82%	8,58%	0%	30,00%
C	0,0321	0,0158	18,76%	4,28%	0,39%	22,73%
D	*	*	*	*	*	*
E	*	*	*	*	*	*
F	0	0	4,55%	7,50%	0%	26%
G	0,0010	0,0040	11,40%	9,75%	0%	28,76%
H	0,0202	0,0177	19,02%	3,22%	3,16%	23,00%
I	*	*	*	*	*	*
J	*	*	*	*	*	*
K	0,0004	0,0020	1,81%	2,26%	0%	8,24%
L	0,0010	0,0040	12,72%	9,88%	0%	30,00%
M	0,0420	0,0370	17,65%	5,32%	1%	22,73%
N	*	*	*	*	*	*
O	*	*	*	*	*	*
P	0,0009	0,0030	2,68%	5,01%	0%	33,68%
Q	0,0050	0,0070	12,09%	9,64%	0%	31,32%
R	0,1067	0,0660	16,86%	6,42%	0%	23%
S	*	*	*	*	*	*
T	*	*	*	*	*	*
U	0,0030	0,0060	3,92%	6,24%	0%	37,27%
V	0,0120	0,0090	11,32%	9,70%	0%	31%
W	0,2170	0,1020	16,40%	6,70%	0%	23%
X	*	*	*	*	*	*
Y	*	*	*	*	*	*

whereas the $w - capacities$ heuristic when using the Gilmore and Gomory [24] method is not able to solve them. Regarding the execution time, both heuristics share similar execution times (see Figure 5).

The solutions obtained through the execution of the $w - capacities$ heuristic presents the best solutions for the copies with $n = 10$ when compared to the solutions obtained in copies with $n = 50$ and $n = 100$, when the solutions are compared , in copies with $n = 10$ the *gap* is close to 5% in the case of 5 classes and below the 5% of *gap* for the categories 20/10, 50/10 and 100/10. Observing the results obtained by Marques [6] it is possible to verify that when the number of items increases, solutions of greater *gap* are obtained (see Figure 4).

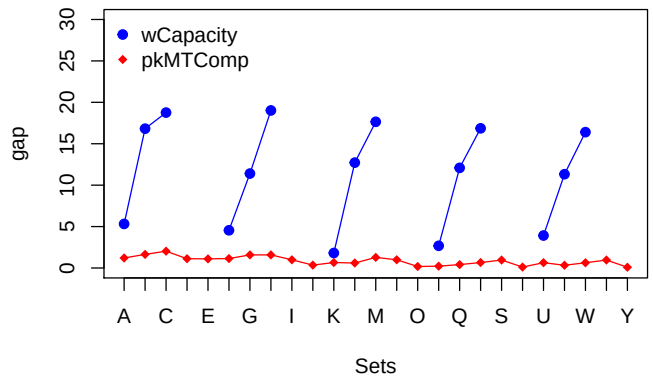

Figure 4: Comparison between Heuristics $w - capacidades$ e $p_kMTComp$ - $\overline{gap}$.

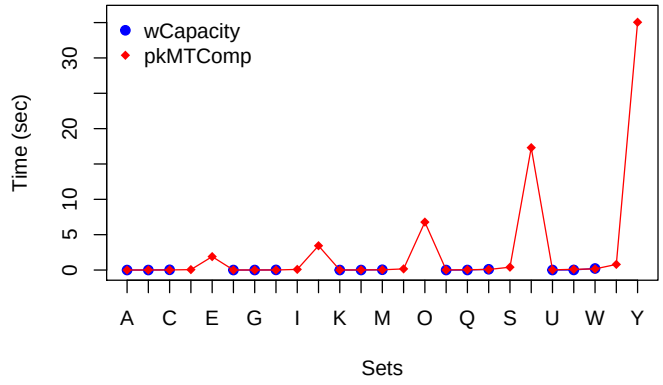

Figure 5: Comparison between Heuristics $w - capacidades$ e $p_kMTComp$ - $\overline{T}$.

4 Conclusion

This work addresses the problem of the compartmentalized knapsack, and presents three new heuristics for its resolution. In the development of two heuristics, the programming language C and *threads* were used, providing a better use of the computational resource and reducing the time of execution of the simulations.

A modification was made in the MTU1 algorithm, to include the knives that are present in the linear model proposed by Inarejos [4], leaving the modeling of the algorithm true to reality, a modification was also made in the Algorithm for elaborating viable compartments, including the utility associated with each compartment in its creation.

As contributions, the $p_kMTComp$ heuristic can be highlighted, where preliminary experiments indicate that the method is promising. The indicative in relation to the execution time shows that when compared with the heuristics p_kX and $p_kGREEDY$ the

method using the algorithm of Martello and Toth [1], presents results with the smaller *gap*, resulting in solutions closer to the optimum without the use of proprietary solutions to solve the subproblems generated by the heuristic.

When compared to the $w - capacities$ heuristic, the tests performed indicate that the $p_kMTComp$ heuristic produces results closer to the optimum, that is. lower *gap*. The $p_kMTComp$ heuristic is capable of solving categories with a higher number of items in each class, which does not happen with the $w - capacities$ heuristic, another detail is in relation to the *gap* obtained with the heuristic $p_kMTComp$ being less than the $w - capabilities$ heuristic. Thus, the execution of numerical simulations and analysis of the results indicate that the $p_kMTComp$ heuristic is superior for solving the compartmentalized knapsack problem, when compared to the $w - capacities$ heuristic.

As future perspectives is the implementation of the heuristic for the restricted case of the Compartmented Knapsack Problem.

References

[1] Silvano Martello and Paolo Toth. *Knapsack Problems: Algorithms and Computer Implementations*. John Wiley & Sons, Inc., USA, 1990.

[2] Hans Kellerer, Ulrich Pferschy, and David Pisinger. *Knapsack Problems*. Springer, Berlin, Germany, 2004.

[3] Robinson S. V. Hoto. *O Problema da Mochila Compartimentada Aplicado no Corte de Bobinas de Aço*. PhD thesis, UFRJ, 2001.

[4] Osvaldo Inarejos, Robinson Hoto, and Nelson Maculan. An integer linear optimization model to the compartmentalized knapsack problem. *International Transactions in Operational Research*, 26(5):1698–1717, dec 2017.

[5] Aline Aparecida de Souza Leão. Geração de colunas para problemas de cortes em duas fases. Master's thesis, ICMC -USP São Carlos, 2009.

[6] Fabiano Marques and Marcos Arenales. The constrained compartmentalised knapsack problem. *Computers and Operations Research*, 34:2109–2129, Jul 2007.

[7] R.W. Haessler. Solving the two-stage cutting-stock problem. *Omega, The International Journal of Management Science*, v 2(n 7):p 145–151, 1979.

[8] E.J. Zak. Modeling multistage cutting stock problems. *European Journal of Operational Research*, 141:313–327, 2002.

[9] J.S. Ferreira, Neves, M.A., and P.F. Castro. A two-phase roll cutting problem. *European Journal of Operational Research*, 44:185–196, 1990.

[10] Robinson Hoto, Nelson Maculan, Fabiano Marques, and Marcos Arenales. Um problema de corte com padrões compartimentados. *Pesquisa Operacional*, v.23(n.1):p.169–187, April 2003.

[11] Robinson Hoto, Alexandre Fenato, Horacio Yanasse, Nelson Maculan, and Fernando Spolador. Uma nova abordagem para o problema da mochila compartimentada. In *XXXVIII SBPO*. SBPO, 2006.

[12] Fabiano do Prado Marques and Marcos Nereu Arenales. O problema da mochila compartimentada e aplicações. *Pesquisa Operacional*, 22(3):285–304, December 2002.

[13] J. M. Valério de Carvalho and A. Rodrigues. An lp based approach to a two-phase cutting stock problem. *European Journal of Operational Research*, 84:580–589, 1995.

[14] Aline Aparecida de Souza Leão, Maristela Oliveira dos Santos, Marco Nereu Arenales, and Robinson Hoto. Uma heurística para o problema da mochila compartimentada. XL SPBO, Set 2008.

[15] John J. Quiroga-Orozco, J.M. Valério de Carvalho, and Robinson S. V. Hoto. A strong integer linear optimization model to the compartmentalized knapsack problem. *International Transactions in Operational Research*, 26(5):1633–1654, feb 2019.

[16] A. Sobhani, A. Yassine, and S. Shirmohammadi. Qoe-driven optimization for dash service in wireless networks. In *2016 IEEE International Symposium on Multimedia (ISM)*, pages 232–237, 2016.

[17] Everton Pereira da Cruz. Uma abordagem heurística linear para mochilas compartimentadas restritas. mathesis, Universidade Estadual de Londrina, 2010.

[18] Robinson Hoto, Fernando Spolador, and Fabiano Marques. Resolvendo mochilas compartimentadas restritas. In *XXXVII SBPO*, 2005.

[19] Aline A.S. Leão, Maristela O. Santos, Robinson Hoto, and Marcos N. Arenales. The constrained compartmentalized knapsack problem: mathematical models and solution methods. *European Journal of Operational Research*, 212(3):455–463, aug 2011.

[20] R. Hoto, N. Maculan, and A. Borssoi. A study of the compartimentalized knapsack problem with additional restrictions. *IEEE Latin America Transactions*, 8(3):269–274, 2010.

[21] *FICO Xpress Optimization* FICO, Jun 2020.

[22] Silvano Martello and Paolo Toth. An exact algorithm for large unbounded knapsack problems. *Operations Research Letters*, 9(1):15 – 20, 1990.

[23] T. Gau and G. Wascher. Cutgen1: A problem generator for the standard one-dimensional cutting stock problem. *European Journal of Operational Research*, 84(3):572–579, 1995.

[24] R. E. , P. C. Gilmoreand Gomory. A linear programming approach to the cutting stock problem—part ii. *Oper. Res.*, 11(6):863–888, dec 1963.