Inteligencia Artificial, 24(67) (2021), 71-89
doi: 10.4114/intartif.vol24iss67pp71-89

INTELIGENCIA ARTIFICIAL

http://journal.iberamia.org/

A fuzzy approach for the variable cost and size bin packing problem allowing incomplete packing

Jorge Herrera-Franklin[1,2,A], Alejandro Rosete[2,B], Milton García-Borroto[2,C]

[1]Maritime Transport Division, Center of Research and Environmental Management of Transport, La Habana, Cuba
[A]franklin@cimab.transnet.cu
[2]Universidad Tecnológica de La Habana José Antonio Echeverría (CUJAE), La Habana, Cuba
[B]rosete@ceis.cujae.edu.cu, [C]mgarciab@ceis.cujae.edu.cu

Abstract. The Variable Cost and Size Bin Packing Problem (VCSBPP) is a known NP-Hard problem that consists in minimizing the cost of all bins used to pack a set of items. There are many real-life applications of the VCSBPP where the focus is to improve the efficiency of the solution method. In spite of the existence of fuzzy approaches to adapt other optimization problems to real life conditions, VCSBPP has not been extensively studied in terms of relaxations of the crisp conditions. In this sense, the fuzzy approaches for the VCSBPP varies from relaxing the capacity of the bins to the items weights. In this paper we address a non-explored side consisting in relaxing the set of items to be packed. Therefore, our main contribution is a fuzzy version of VCSBPP that allows incomplete packing. The proposed fuzzy VCSBPP is solved by a parametric approach. Particularly, a fast heuristic algorithm is introduced that allows to obtain a set of solutions with interesting trade-offs between cost and relaxation of the original crisp conditions. An experimental study is presented to explore the proposed fuzzy VCSBPP and its solution.

Keywords: Heuristic Search and Optimization, Variable Size and Cost Bin Packing Problem, Fuzzy Systems, Incomplete Packing

1 Introduction

The Variable Cost and Size Bin Packing Problem (VCSBPP) is a generalization of the Bin Packing Problem described in Crainic *et. al.* [5]. VCSBPP consists in packing a set of items in a set of heterogeneous bins (i.e. with different sizes) minimizing the cost of all used bins. For each size, it is assumed an inexhaustible supply of bins. This optimization problem has many practical applications in several fields, such as industrial process [1] and informatics[16].

In general, optimization models are based on constraints and assumptions which may be hard to be satisfied in the real world. One way to add flexibility in optimization models is to introduce fuzzy concepts which is a very active field of research. Examples of active topics in fuzzy optimization from the theoretical point of view are the study of the convergence of the solution methods[14] and the introduction of fuzzy mechanisms to guide the optimization procedure in order to reduce the complexity[21] (see[8] for an extended and updated view of fuzzy sets concepts).

From the practical point of view, several fuzzy optimization approaches have been recently presented to deal with the real-world conditions in diverse fields, such as: uncertain payoffs in the context of incomplete-information games[6], epistemic and aleatory uncertainties in the context of topology (structural) optimization[19], uncertainty of device parameters and about the network structure (failure and changes) for clustering optimization of wireless ad-hoc networks [20], and uncertainty associated to durations and flexible due dates in job shop scheduling [25]. Recent fuzzy approaches for logistic optimization problems are: the consideration of the uncertainties associated to the cost of stations, demands, prices and distance capacity of each drone in order to minimize the cost of an aerial (drone-based) delivery system[23], the management of an uncertain radius of coverage in location problems[10] and the use of fuzzy numbers to manage uncertainties associated to service time, energy consumption, travel time and recharge in the context of optimizing electric vehicle routing problem [31].

Particularly, the uncertainty in the Bin Packing Problem has been addressed from different angles. The first example found was proposed by Kim *et. al.*[18] and consists in relaxing the dimensions of the items to pack, using for that purpouse triangular fuzzy numbers and the reduction cost by a quadratic function. A real-life application appears in Wang *et. al.* [27] where it is described a chance-constrained model where the item sizes are uncertain. In that paper, the bin capacity constraints are satisfied with a given probability. Following this idea (fuzzifying the items area), Eliiyi and Nasibov[9] present a practical application in telecommunications that allows that each item can be packed partially. This work deals with the allocation of data transmission from a base station to many mobile stations for certain frames of time with varying objectives using fuzzy area definitions to allow partial packing among frames, considering item priorities and service quality; this method is very similar to [18]. Although those papers didn't deal with incomplete packing explicitly, it can be understood that by adding fuzziness on item weights, not all items are packed or at least, not all the whole items are packed. In this sense, another example is provided by Crainic *et. al.* [4] that present a real-life application in logistics with uncertainty on the characteristics of the items. Here, the bins included pre-existing capacity plan, are chosen in advance without the exact knowledge of which items will be available for the dispatching. When, during the operational phase, the planned capacity is not enough, extra capacity must be purchased. More recently, Herrera-Franklin *et. al.*[13] tackles a fuzzy approach of the VCSBPP relaxing the constraint related to the capacity of the bins, which might be considered an alternative to the aforementioned works.

In VCSBPP original model all items must be packed. However, as it could be assessed in the mentioned researches, in real-life situations is not always possible (or convenient) to fulfill this condition due to a limited budget for packing. This is currently solved by purchasing more bins or modifying the capacity of them. In this sense, the major issue appears when none of that is possible and only a certain number of items can be packed. An incomplete packing (i.e. a solution where some items are not packed) may be considered a relaxed solution to the original problem. To the best of our knowledge, the situation where incomplete packing is admissible for a decision maker has not been addressed before. Thus, it is impossible to evaluate relaxed solutions (where some items are not packed) even if they allow a notable reduction in cost.

This is the focus of this paper. Here, a fuzzy version of the VCSBPP that allows incomplete packing is introduced, based on the parametric approach [26] and an heuristic algorithm to solve it. The model and the algorithm are explored in an experimental study (with 18 crisp instances with 25 to 100 items, that derive in several relaxed instances for each).

The paper is organized as follows. Section 2 presents the VCSBPP fuzzy model with partial packing. Section 3 presents an illustrative example of the fuzzy VCSBPP. A new method to explore the solutions is explained in section 4. An experimental study is presented in section 5 and discussed in section 6. In section 7, conclusions are made from the analyzed results.

2 A Fuzzy Model for the VCSBPP allowing partial packing

Before introducing the fuzzy extension of the VCSBPP, here we present the classical formulation used in previous works [3, 11, 17]. The model is presented in terms of the following notation:

$I = \{1 \dots n\}$	set of items
w_i	weight of the item $i \in I$
$J = \{1 \dots m\}$	set of bins
W_j	capacity (or size) of the bin $j \in J$
C_j	cost of the bin $j \in J$
x_{ij}	binary variable: 1 if item i is packed in bin j; 0 otherwise
y_j	binary variable: 1 if bin j is used; 0 otherwise

Then, the mathematical model of VCSBPP is as follows:

$$\textbf{\textit{minimize}} \rightarrow \sum_{j \in J} C_j * y_j \tag{1}$$

$$s.t. \sum_{j \in J} x_{ij} = 1, \qquad \forall i \in I \tag{2}$$

$$s.t. \sum_{i \in I} w_i * x_{ij} \leqslant W_j * y_j, \qquad \forall j \in J \tag{3}$$

$$x_{ij} \in \{0,1\}, \quad \forall i \in I,\ \forall j \in J \tag{4}$$

$$y_j \in \{0,1\}, \qquad \forall j \in J \tag{5}$$

The objective function (1) is to minimize the cost of the bins used for packing all the items. Constraint (2) ensures that each item i is packed in one and only one bin, hence, items are not divisible and each one should be packed. Inequality (3) states that (for each used bin j) the sum of the weights of the packed items can not exceed the capacity of the bin used; (4) represent if the item i is packed in bin j (1 if packed, 0 otherwise) and (5) also binary, represents of the bin j is used in the solution (1) or not (0). In VCSBPP there are different types of bins. Here we use Z to denote the number of different values of W_j.

In order to present the proposed fuzzy model for VSCBBP, we introduce the following examples:

- **Example 1.** There are two items ($I = 2$) with weights $w_1 = 99$ and $w_2 = 3$ to be packed in three bins ($J = 3$) of only one type ($Z = 1$, capacity $W_j = 100$, costs $C_j = 10$ for $j = 1,2,3$). For this example, one optimal solution s_A is to place each item in separate bins as it is shown in Figure 1 (e.g. by assigning $x_{11} = 1$, $x_{21} = 0$, $x_{12} = 0$, $x_{22} = 1$, $x_{13} = 0$, $x_{23} = 0$) because the bin capacity $W_j = 100$ does not allow to pack both items together because of their combined weights ($w_1 + w_2 = 99 + 2 = 101 > 100$). This solution s_A satisfies all conditions and it reaches a cost of 20, because it uses two bins (each one with cost $C_j = 10$). It is worth noting that a solution s_B that only packs the first item ($x_{11} = 1, x_{21} = 0, x_{12} = 0, x_{22} = 0, x_{13} = 0, x_{23} = 0$) has a cost of 10 (because only one bin is used) but this solution s_B is unfeasible because the condition $\sum_{j \in J} x_{ij} = 1$ does not hold for all items (item $i = 2$ is not packed) and, consequently, the universal quantifier ($\forall i \in I$) in constraint (2) is not satisfied. Also, there is another solution s_X that only packs the second item and it may be comparable to the unfeasible solution s_B (both have a similar cost of 10 and they are unfeasible because one item is not packed). However, it seems that this solution s_X is not preferable to s_B because of the difference in terms of the packed weight.

- **Example 2.** There are 34 items ($I = 34$) with weights $w_i = 3$ for $i = 1..34$ to be packed in three bins ($J = 3$) of one type ($Z = 1$, capacity $W_j = 100$, costs $C_j = 10$ for $j = 1,2,3$). For this example, an optimal solution s_C is to place all but one item ($i = 1..33$) in the first bin and the final item ($i = 34$) in the second bin (see Figure 1). As in **Example 1** this solution s_C satisfies all conditions and reaches a cost of 20. Again it is worth noting that a solution s_D consisting in a variation of s_C that avoids packing the item $i = 34$ in the second bin (i.e. by changing $x_{34,2}$ from 1 to 0) has a cost of 10 but this solution s_D is unfeasible because the universal quantifier ($\forall i \in I$) in constraint (2) is not satisfied (in this case, for $i = 34$). In this **Example 2**, there is another optimal solution s_E (with equal cost of 20) also represented in Figure 1. Here, 2 bins are used too but, this solution consists of placing 17 items with weight $w_i = 3$ in each bin. In strict terms, this solution s_E is as good as the solution s_C because they are feasible and they have a cost of 20. In this case, if the second bin of solution s_E is not used, an unfeasible solution s_Y with cost of 10 is reached. However, in this solution s_Y the universal quantifier ($\forall i \in I$) in constraint (2) is not satisfied for $i = 18..34$ because 17 items are not packed.

In practical situations, a decision-maker could prefer a solution that packs *almost all* the total weight if this implies a notable reduction in cost. **Example 1** and **2** illustrate this point because the second solutions described (s_B and s_D, respectively) pack 97% of the total weight (99 out of 102) with a reduction of the 50% in cost (10 out of 20) with respect to the optimal solutions (s_A, s_C, s_E). This trade-off must be considered by a decision maker under conditions that restrict the loading capacity. This concern is not taken into account in the crisp formulation of VCSBPP because it forces to pack all items, thus making unfeasible the relaxed solutions s_B and s_D.

The fuzzy version of VCSBPP, introduced in this paper, considers that partial-packing solutions (such as s_B and s_D) are feasible with a certain degree, i.e. it considers constraint (2) as a fuzzy constraint that is satisfied to a certain extent that depends on the percent of the total weight that is packed in the solution at hand.

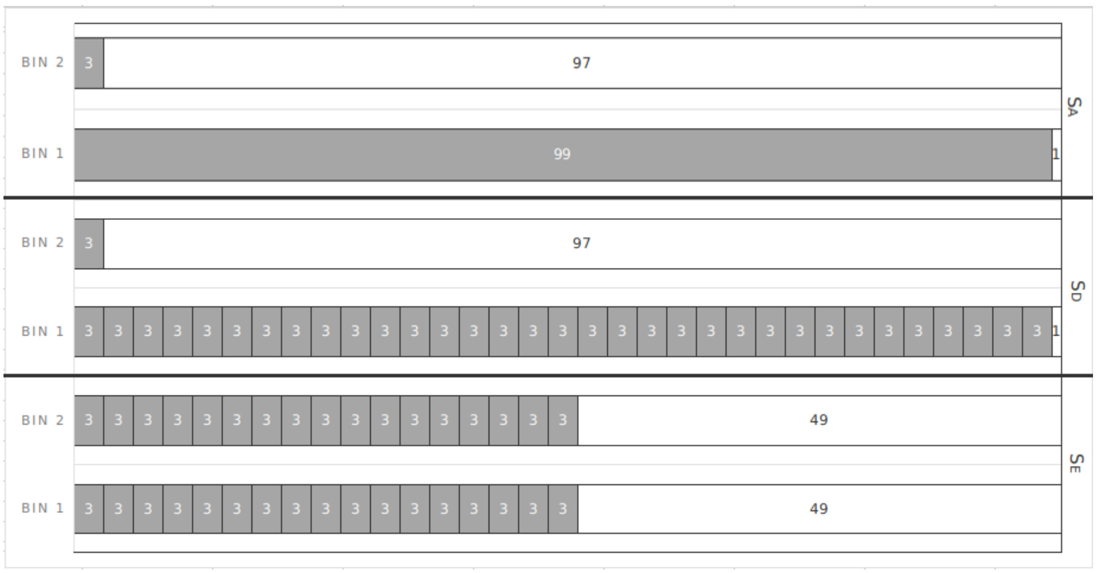

Figure 1: Optimal solutions s_A, s_C and s_E of **Example 1** and **Example 2**

The usefulness of the fuzzy interpretation of the universal quantifier ($\forall$) is a very important field in fuzzy theory, very related to the fundamental works on data summarization conducted by Yager [29, 30] and Kacprzyk [28], and it is a very active field today [2, 15] that it is impossible to fully describe here. In particular, allowing intermediate meanings such as *few* or *most* instead of the extreme logical quantifiers (*exists* and *for all*) is very important in summarizing databases where the classical (extreme) quantifiers are unlikely to be satisfied. This fact makes convenient the use of relative quantifiers [22] that measure the proportion of satisfying cases with respect to the total number of cases, thus assigning a fuzzy meaning to imprecise concepts (e.g. *minority/majority*, *most/little of*). In these situations, each unit/case (that is considered in order to define the proportion to be converted in a degree of membership) may be an example/tuple in the database.

In our case, we identify two possible units to relate to the membership of a partial packing: the number of items packed with respect to $|I|$; or the amount of weight packed with respect to the total weight of all items I. We opt for the second option as in general terms it could be more adequate to express the meaning of a partial fulfillment of constraint (2). In this way, we are assuming the use of a relative quantifier *most* that must be non-decreasing and with extreme values of 0 and 1 in order to be coherent [2].

Particularly, here we are considering that the universal quantifier ($\forall$) may be substituted for a fuzzy quantifier ($\tilde{\forall}$) meaning that *almost all the total weight* is packed, as a particular case of the relative quantifier *most* [2]. In the homogeneous case with respect to the item types (i.e. all items have equal weight, as it is the case of **Example 2**) the previous meaning matches with *almost all items*, but in general is preferable to take into account the total weight that it is more related to the focus of the problem. This is in line with the natural preference for solutions such as s_B and s_D over solutions such as s_X and s_Y in **Examples 1** and **2**. The fuzzy version of the constraint (2) is presented in constraint (6).

$$s.t. \sum_{j \in J} x_{ij} = 1, \quad \tilde{\forall} i \in I \tag{6}$$

Let U denote the fuzzy set associated to a solution s that means "*the condition $\tilde{\forall}$ in constraint (6) is satisfied*". For a particular solution $s \in S$ (the solution space) the degree of membership μ_U to the set U, may be defined as $\mu_U = \mu(p_s)$, where p_s is the proportion of the total weight that is packed in the solution s as it is defined in expressions (7)-(9). In the previous examples, $p_{s_A} = 1$, $p_{s_B} = 0.97$, $p_{s_C} = 1$, $p_{s_D} = 0.97$.

$$t_s = \sum_{i \in I} w_i \sum_{j \in J} x_{ij} \tag{7}$$

$$T = \sum_{i \in I} w_i \tag{8}$$

$$p_s = \frac{t_s}{T} \tag{9}$$

There are several ways to define the function $\mu(p_s)$ that expresses the degree of membership μ_U for a particular situation. An example is to use $\mu(p_s) = p_s$, i.e., it depends on the proportion p_s that represents the total weight of the items packed in solution s with respect to the total weight of all items computed in (8).

As $p_s \in [0,1]$ in (9) it may be considered a candidate to be a membership function μ_U, but there could be more options, as it is shown in Figure 2. The case $\mu(p_s) = p_s$ corresponds to the function Proportional in Figure 2. Other alternatives may be generalized using trapezoidal functions, as in expression 10, for defining the degree of membership [8].

$$\mu_(p_s) = \begin{cases} \frac{p_s - a_1}{a_2 - a_1}, & a_1 < p_s < a_2, \\ 1, & a_2 \leqslant p_s \leqslant a_3, \\ \frac{a_4 - p_s}{a_4 - a_3}, & a_3 < p_s < a_4, \end{cases} \tag{10}$$

In terms of the parameters (a_1,a_2,a_3,a_4) as $p_s \leqslant 1$, acording to (9) the last case of (10) is never used. It is also worth noting that $a_3 = a_4 = 1$ in order to guarantee that μ_{p_s} is coherent with Figure 2 being non-decreasing. In this sense, the functions presented in Figure 2 are defined as follows: Crisp (1,1,1,1), Proportional (0,1,1,1), V_1 (0.6, 0.9, 1,1) and V_2 (0.7, 1,1,1), where the x-axis corresponds to p_s as defined in (9). Crisp represents the original VCSBPP problem where the (strict) feasibility condition is to pack all items. The other functions are different ways to understand the incomplete packing with different degrees of membership. For example, in the Proportional function, the membership is directly related to the proportion of the total weight included in the solution. In addition to the classical Crisp and the Proportional functions, other functions may allow alternative schemes to define the feasibility. For example, the use of V_1 means that a decision-maker considers that a solution that packs at least 90% of the total weight T is as feasible as the solution to the crisp problem that packs all items. It is worth noting that all functions in Figure 2 are coherent [2] with the implicit *most* relative quantifier, i.e. they are non-decreasing with extreme values of 0 and 1.

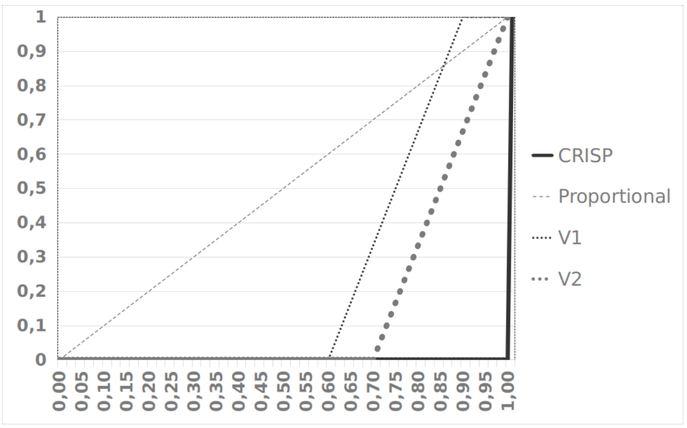

Figure 2: Functions to define membership

If we apply this fuzzy approach to **Example 1** (respectively, **Example 2**) by using V_1 membership function (Figure 2), solution s_B (respectively, s_D) is feasible with a degree of 1 and consequently it is as feasible as solutions s_A (respectively, s_C). Thus s_B (respectively, s_D) is the optimal one for **Example 1** (respectively, **Example 2**), as it is better that solution s_A (s_C) in terms of the objective function (1). If the Proportional function is considered, the degree of membership s_B and s_D is 0.97.

To tackle this situation we rely on the parametric approach [26]. This implies that, associated to a fuzzy problem there are several crisp instances defined by different $\alpha - cuts$ (i.e. only those solutions with a feasibility greater than or equal to α are considered feasible [8, 10]). For example, with the Proportional membership function (Figure 2), solutions s_B and s_D are feasible with a degree of 0.97 and hence they are unfeasible if $\alpha = 0.99$ but they are feasible if $\alpha = 0.95$. This implies that in **Example 1** the solution s_A (with cost of 20) is the optimum if $\alpha = 0.99$ but the optimal solution is s_B (with cost of 10) if $\alpha = 0.95$. Thus, the solution to a fuzzy problem according to the parametric approach consists of a set of optimal solutions associated with different values of α.

With this in mind, in (11) we define the set S^α of feasible solutions for a certain value of α with respect to the fuzzy constraint (6) in terms of the total weight packed by each solution (7).

$$S^\alpha = \{s \in S \mid \mu(p_s) \geqslant \alpha\} \tag{11}$$

According to (11) for the particular case of the Proportional function (Figure 2), a solution s is considered α-feasible if it satisfies condition (12).

$$p_s \geqslant \alpha \tag{12}$$

Finally, it is worth noting that in **Example 1** the solution s_X that only uses the second bin ($x_{11} = 0$, $x_{21} = 0$, $x_{12} = 0$, $x_{22} = 1$, $x_{13} = 0$, $x_{23} = 0$) has a cost of 10 (only one bin is used) but this solution s_X is unfeasible with $\alpha = 0.95$ because it only packs $t_s = 3$ and therefore $p_s = 0.03$ (according to Proportional function), i.e. condition (12) does not hold ($0.03 \ngeqslant 0.95$).

3 Solving fuzzy VCSBPP with partial packing

This section illustrates how to obtain a fuzzy solution to the introduced fuzzy VCSBPP that allows partial packing, based on a parametric approach similar to the one presented in several previous papers, e.g. [10, 13, 26]. This scheme consists in setting different values of α thus producing different crisp instances that are solved to obtain the fuzzy solution (a set of crisp solutions for each particular α).

It is noteworthy that in several problems [10, 13, 26] it is hard to note the effect of the different values of α. For example, Herrera-Franklin *et. al.* [13] presents a fuzzy version of VCSBPP that allows bin overloading thus, increasing the amount of weight to be packed in each bin, but the real impact in the solution needs to be evaluated. In general, it is convenient to explore interesting values of α to produce significant trade-offs between relaxation of the original condition and cost (see an example in the context of routing problems in [24]).

The same may occur in our case with a particularity. Here we know that some values of membership are impossible to reach. For example, there is no solution with membership equals to 0.01 in **Example 1**. Indeed, as in this example there are only two items to be packed, there are only 4 possible values of membership that are interesting: the case associated to the original situation where both items are packed, the case where only is packed the item 1 with $w_1 = 99$, the case where only is packed the item 2 with $w_2 = 3$, and the extreme case where no items packed. The associated membership are 1, 0.97, 0.03 and 0, respectively. Thus, it makes no sense to try with values of α such as 0.4 or 0.8. In general, there are (at most) $2^{|I|}$ values of membership that are interesting, those related to the different subsets $\tilde{I} \subset I$ associated with the different partial packings. This situation is analyzed in the following example.

- **Example 3.** The original crisp instance of this example consists of three different bin types ($Z = 3$) with capacities $W_j \in \{50, 100, 150\}$, the cost function is linear with the form $C_j = 15 * W_j + 32$ and there are 25 items ($|I| = 25$) with weights w_i randomly drawn following a Uniform Distribution in the interval $[20, 120]$. It is available in `http://www.cimab.transnet.cu/files/Example3FVCSBPP.zip`.

In this instance, the total weight is $T = 1796.74$ computed as it is described in (8). Figure 3 presents the solutions that corresponds to the (relaxed) instances corresponding to the cases of set of items $\tilde{I}$ where only one item is not packed, i.e. $|\tilde{I}| = |I| - 1$. The feasibility of each solution considering the Proportional membership is shown in the x-axis and the corresponding cost in the y-axis. Each instance was solved by using CPLEX based on the integer linear programming formulation of VCSBPP. The 25 relaxed solutions and the crisp (original) case where the 25 items are packed are included in Figure 3.

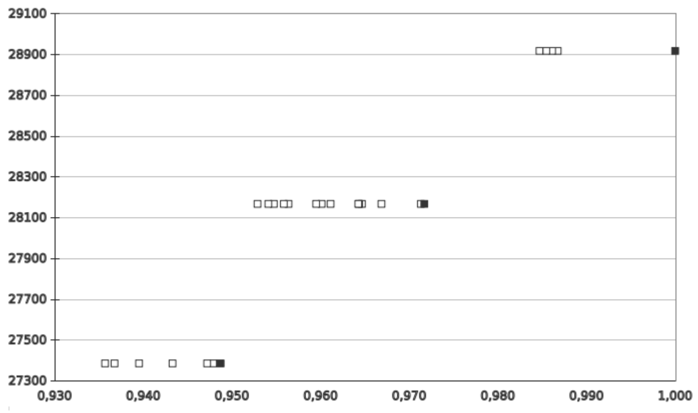

Figure 3: Optimal solutions for the relaxations associated with packing all but one bins

These 26 instances are a very small sample of the whole set of possible relaxations of **Example 3**, because there are $2^{25} = 33554432$ partial packings of 25 items. In spite of this fact, Figure 3 illustrates several important aspects.

- It clearly shows the trade-off between cost and feasibility. More that 600 units of cost may be saved by considering $\alpha = 0.97$, and more than 1400 units of cost if $\alpha = 0.94$.
- The exploration of different cases based on removing only one item produces a limited spectrum on the relaxations (all feasibilities are above 0.93). So, if a greater reduction in cost is needed, more relaxed cases are to be considered.
- There are only three different values of cost, thus only three solutions are really interesting for a decision maker (black dots in Figure 3), because it seems unlikely to prefer a more relaxed version (white dots) if it does not allow a reduction in cost. In this paper we use the term ”dominated” to refer to these solutions with the same cost and less feasibility (i.e. white dots in Figure 3) in spite of the fact that the feasibility is not strictly an objective, but a constraint. We use this metaphor here to express the inner rationality of the situation.
- There are some interesting cases not explored with the previous scheme, such as removing of two or three items.

In general, this poses the problem of how to better sample the space of possible relaxations, taking into account that the complete space of relaxations is too large, e.g. $2^{|I|} = 33554432$. In addition, the necessity to solve each generated problem is a considerable effort to be faced (i.e. in this case the 25 relaxed instances must be solved by using CPLEX).. Both aspects may be faced by taking a different approach. The previous scheme is based on first defining each instance and then solve it, by a general solution procedure. Instead of that, the optimal solution of the crisp instances may be used as a baseline solution to generate different relaxed solutions directly, with their corresponding cost and feasibility. This new approach is presented in the next section.

4 Method for exploring the trade-off between cost and feasibility in fuzzy VCSBPP with partial packings

In this section, we introduce a new method to obtain the fuzzy solution of VCSBPP that explores only the cases where the reduction of the items to pack has a sure impact in the cost. It starts from the solution associated to the non-relaxed version of the problem represented in a particular optimal packing containing the structure of this solution.

The idea is to consider every possible removal of each bin (instead of removing items) in the solution. It is worth noting that given a solution (optimal or not) for a crisp original instance its is possible to obtain a (several) solution(s) by reducing the bins that are used (i.e. to free some of them). Indeed, the initial **Example 1** and

Example 2 illustrated this transformation, thus producing new solutions (such as s_B and s_D) with different cost and feasibility. The cases discussed in **Example 3** are not in that line, because in this case items are removed (instead of bins). Indeed, in **Example 3** the structure of the optimal solution for the crisp case, in terms of the used bins, is not considered at all. This implies that despite 26 instances of the problem are solved, only three interesting trade-offs between cost and feasibility were obtained.

Before explaining the proposal, it is useful to conveniently prepare the available data by following expressions (13)-(17). First, it is obtained the set J^* of bins that are really used in the solution (13). Then, these bins are partitioned (14) according to their types ($z = 1...Z$) in several sets J^z. For each bin, the total weight packed in the j^{th} bin fo type z is computed and stored in a matrix P, that can be access as $P[z, j]$ as it is expressed in (15) and the number of bins used of each type are stored in the array B as in (16). For each type, bins are sorted in J^z in ascending order according of the their total weight packed P as it is indicated in (17).

$$J^* = \{j \in J| \qquad \sum_{i \in I} x_{ij} > 0\} \tag{13}$$

$$\bigcap_{z=1..Z} J^z = \emptyset, \qquad \bigcup_{z=1..Z} J^z = J^* \tag{14}$$

$$P[z, j] = \sum_{i \in \tilde{I}} w_i * x_{ij}, \qquad \forall j \in J^z \tag{15}$$

$$B[z] = |J^z|, \qquad \forall z \in 1,2...Z \tag{16}$$

$$P[z, j] \leqslant P[z, j+1], \quad \forall z \in 1,2...Z, \forall j \in J^z \tag{17}$$

Now, the main idea of the proposed algorithm to obtain a set of relaxed solutions is presented. The intention is to explore a small fraction of the possible relaxations, based on a rational sequence for sampling the cost/feasibility trade-off. First, consider the case where the optimal solution only uses one bin type (it is worth to note that it is possible that this occurs in the optimal solution, even when there are several available types). Assume this homogeneous solution (equivalent to the situation where $Z = 1$) and that bins are sorted in ascending order according to their total weight, as it is defined in (17). In this situation, it is rational to think in to free the first bin (with the less weight), because freeing this less weighted bin produces the less reduction in the feasibility among all the solutions with the same cost. This reasoning is in line with the preference for solution s_B over solution s_X in the **Example 1**.

The same applies for the the relaxation associated to free the two (or more) less weighted bins with respect to other combination of two (or more) bins. In general for the homogeneous case that employs $B[z]$ items of the same type z, its is only interesting to consider the $B[z]+1$ solutions obtained by freeing the first 0,1, ... $B[z]$ bins (the less weighted ones, according to $P[z, j]$. If the solution for the crisp case uses several types of bins ($B[z]$ indicating the number of used bins for each type z, $z = 1, ...Z$), the same rationale applies to only consider the solutions based on the combinations of freeing the first (less weighted) bins of each type. This number of solutions is determined by (18).

$$\prod_{z=1..Z} [B[z]+1] \tag{18}$$

For example, if the optimal solution for the crisp problem with 25 items consists of 3 bins of type 1 with $W_j = 50$ (i.e, $B[1] = 3$), 6 bins of type 2 with $W_j = 100$ (i.e, $B[2] = 6$), and 4 bins of type 3 with $W_j = 150$ (i.e, $B[3] = 4$), then the number of solutions to be explored according to (18) is $(B[1]+1)*(B[2]+1)*(B[3]+1) = 4*7*5 = 140$. In general $\sum_{z=1..Z} B[z] \leqslant |I|$ because no more than $|I|$ bins are needed to pack $|I|$ items. The number of solutions (18) generated by the previous approach becomes greater when the number of bins used is similar for each type ($B[z] \approxeq \frac{|I|}{Z}$). Even in this situation, the numbers of solutions to be analyzed is considerably less than $2^{|I|}$. For example, for $|I| = 25$ and $Z = 3$ we obtain $B[z] \approxeq \frac{25}{3} = 8.33$, thus, considering the impossible situation $B[z] = 9$ (for $z = 1,2,3$) we have to explore $10^3 = 1000$ solutions instead of the $2^{25} = 33554432$ possible subsets $\tilde{I}$. Nevertheless, a worst case analysis of this algorithm is presented in section 5.1.

In addition, the solutions obtained by the proposed method may be directly evaluated without executing the solver each time. The structure of each relaxed solution is also directly obtained because the items that are not packed and the bin that are free are explicitly obtained. Thus, it is direct to obtain their costs and degrees of feasibility. This is similar to a bi-objetive problem that tries to minimize the cost (by removing bins) and the membership. Algorithm 1 presents the main idea previously explained.

```
Data: $B$, $P$, $C$, $P_{max}$
Result: $Q, M, S, R$
1  $s = 0$;
2  $f \leftarrow \emptyset$;
3  $Q, M, S, R \leftarrow \emptyset$;
4  while not $f[1] = -1$ do
5      $s \leftarrow s + 1$;
6      $S[s] \leftarrow B - f$;
7      $Q[s] \leftarrow C \cdot S[s]$;
8      $CNotPacked \leftarrow 0$;
9      forall $z \in Z$ do
10         for $i \leftarrow 0$ to $f[z]$ do
11             $CNotPacked \leftarrow CNotPacked + P[z, i]$;
12     $M[s] \leftarrow 1 - (CNotPacked / P_{max})$;
13     $R[s] \leftarrow RectifyMembership(s)$;
14     $f \leftarrow next(f, B, Z)$;
```

Algorithm 1: . Heuristic to obtain relaxed solutions from the CRISP solution

The idea of Algorithm 1 is to collect all possible relaxations related to the freeing of a certain amount of bins of each type. In Algorithm 1, each element $f[z]$ of the array f indicates that items, originally packed in the lest weighted $f[z]$ bins of type z, now are not packed, i.e., the number of the less weighted bins that are being freed. The structure of each solution s is obtained in step 6, and its cost and relaxation are computed in steps 7 and 12, respectively. The function *next* in step 14 is used to systematically obtain each possible relaxation. This function is described in Algorithm2. To clarify the idea of this algorithm, it can be assumed the the optimal solution obtained by CLPEX for the **Example 3**, that consists in $B[z] = [0, 1, 12]$. For this example, the function *next* retuns the following sequence of values of f: $[0,0,0]$, $[0,1,0]$, $[0,0,1]$, $[0,1,1]$, $[0,0,2]$, $[0,1,2]$,...$[0,0,11]$, $[0,1,12]$.

```
Data: $f$, $B$, $Z$
Result: $f$
1  $z = 0$;
2  $found$ = false;
3  while not $found$ and $z \leqslant |Z|$ do
4      if $f[z] \leq B[z]$ then
5          $f[z] \leftarrow f[z] + 1$;
6          $found \leftarrow$ true;
7      else
8          $f[z] \leftarrow 0$;
9          $z \leftarrow z + 1$;
10 if not $found$ then
11     $f[z] \leftarrow -1$;
```

Algorithm 2: . Function to explore all possible relaxations

The Algorithm 1 uses the matrix P and the array B according to the expressions (15) and (16). As bins are grouped by types, their cost are stored in the array C, i.e. the cost is considered for each type ($C[z]$), instead of considering it for each particular bin (C_j). The total weight of the crisp (original) solution obtained from (8) is assumed to be precalculated in the variable P_{max} as in expression (19).

$$P_{max} = \sum_{z \in Z, j \in J^z} P[z, j] \tag{19}$$

Also, as it was aformentioned, it is defined the array f parallel to B where each $f[z]$ indicates the number of bins of each type z that are free in the currently analyzed solution and *CNotPacked* stores the cumulative weight that is not packed associated to each type in the current solution. It is worth noting that $M[s]$ is computed based on the Proportional function of Figure 2.

The set of s solutions obtained by Algorithm 1 are returned in the arrays Q and M, in terms of their costs and membership/feasibility, respectively. Notice that $S[s]$ is obtained by a rest of vectors B (structure of the crisp solution) and the current structure f while $Q[s]$ is the dot product between the cost of the crisp solution C and the current solution $S[s]$, because in both cases it is related to the number of bins by type $f[z]$ included in the solution. Those values are calculated in each iteration, then two nested cycles are used (lines 9-13) to accumulate the freed space and with that value, to obtain the membership degree $M[s]$. After this step, it is obtained $R[s]$ that is a rectified value of membership that will be explained in the last paragraph of this section. The structure of each solution (number of bins by type) is also obtained in the matrix S that is accessed as $S[s,z]$ for types $z = 1,2,3$, i.e. the value $S[s,z]$ expresses the number of bins of the type z that are used in the solution s.

The results of the Algorithm 1 for the **Example 3** are presented in Figure 4 and Table 1. For this example, the algorithm started from the optimal solution of the crisp instance obtained by CPLEX where $B[z] = [0,1,12]$. The trade-off between membership (x-axis) and cost (y-axis) is graphically shown in Figure 4

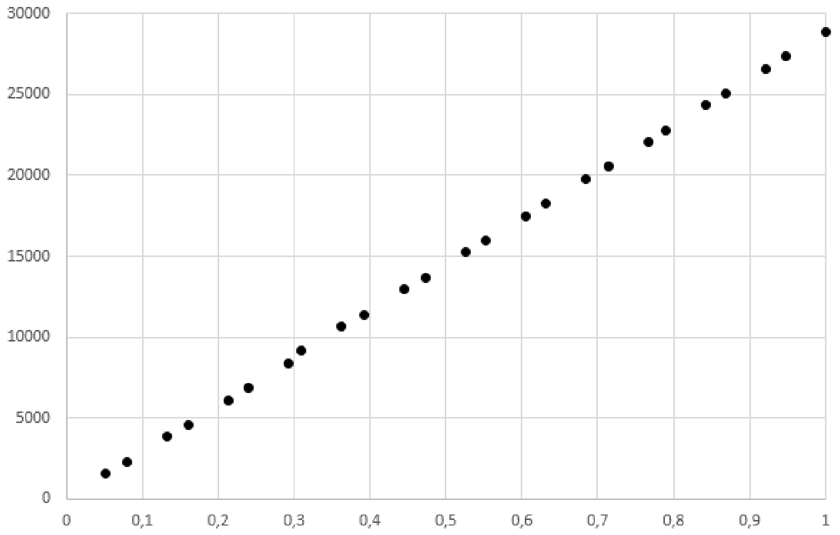

Figure 4: Trade-off between $R[s]$ and $Q[s]$ in the solutions obtained for **Example 3**

Table 1 presents all solutions for **Example 3** with their corresponding cost $Q[s]$, membership $M[s]$, and the number of bins of each type that it uses $S[s,z]$. They are sorted by cost $Q[s]$ in descending order. The trivial solution with no bins, no cost and null feasibility is ignored. Notice, how the structure of the solutions varies, provoking singular cost/membership trade-offs as part of the fuzzy solution of problem.

With these set of solutions (that represents a solution to the fuzzy problem), a decision maker can decide based on their real conditions. For example, if she/he desires that at least 80% of the weight must be packed ($M[s] \geqslant 0.8$) the cost may be at least 24352. On the other hand, if the available budget is 20000 ($Q[s] \leqslant 20000$), the maximum weight that must be packed is 68%*T according to the solutions in Table 1.In the first case, the value of $\alpha = 0.8$ defines that the solutions behind this value of membership are not α-feasible, so the best cost among the available ones may be considered the optimal. In addition, some consideration about the structure of the solution may be taken into account (for example, an homogeneous solution may be desirable in certain situation, whereas the minimum number of used bins ($S[s,1]$ + $S[s,2]$ + $S[s,3]$) may be preferable in another situation.

It is worth noting that the feasibility $M[s]$ obtained in Algorithm 1 may be improved later in terms of membership (feasibility). That is the reason of the value $R[s]$ and the call to the function *RectifyMembership* in Algorithm 1. Resuming **Example 2**, if the optimal (crisp) solution s_E with 17 items in each bin is modified by using Algorithm 1, it is obtained the (relaxed) solution s_Y with one bin, with cost of $Q[s] = 10$ and membership $M(s) = 0.5$ (only 50% of the total weight is packed, i.e. 17 items). However, this solution s_Y may be improved by using the function *RectifyMembership* to reach the solution s_D described in **Example 2** without affecting the cost $Q[s] = 10$, but with increased feasibility of $R[s] = 0.97$.

$Q[s]$	$M[s] = R[s]$	$S[s,1]$	$S[s,2]$	$S[s,3]$
28916	1	0	1	12
27384	0,947	0	0	12
26634	0,922	0	1	11
25102	0,869	0	0	11
24352	0,842	0	1	10
22820	0,789	0	0	10
22070	0,767	0	1	9
20538	0,715	0	0	9
19788	0,685	0	1	8
18256	0,632	0	0	8
17506	0,605	0	1	7
15974	0,552	0	0	7
15224	0,526	0	1	6
13692	0,473	0	0	6
12942	0,445	0	1	5
11410	0,392	0	0	5
10660	0,362	0	1	4
9128	0,309	0	0	4
8378	0,292	0	1	3
6846	0,239	0	0	3
6096	0,213	0	1	2
4564	0,160	0	0	2
3814	0,132	0	1	1
2282	0,079	0	0	1
1532	0,052	0	1	0

Table 1: Solution obtained by Algorithm 1 for **Example 3**

RectifyMembership is a simple heuristic inspired by First Fit Decreasing heuristic [7] that consist in sorting the non-packed items in descending order by their weights and inserting them in the first bin where they fit. If it is possible to insert some items (as in the previous example), the membership is updated to obtain $R[s]$. The relative importance of this rectification is connected to the inner strategy used in each solution method (e.g. Simplex method in CPLEX) with respect to the placement of items in bins when several optimal solutions are available with the same cost.

In the results presented in Table 1 the value of $R[s]$ is inserted in the same column of $M[s]$ because the rectification does not produce any improvement in this example. This aspect will be analyzed in more detail in the experiments conducted in the next section. In general, as bins are freed in ascending order of the associated decay of feasibility in Algorithm 1, we consider that the importance of making the rectification is attenuated by our strategy.

In this regard, for each solution of the Table 1, the optimal solution for the corresponding relaxed instances (by freeing all the items in the bins in f) was obtained by using CPLEX. These solutions are also included in the *url* of the **Example 3** in section 3, being identified in the filename with the α value. In all cases, the optimal solution obtained by CPLEX is identical to the one obtained by Algorithm 1, but there is no guaranty of optimality because of the heuristic nature of Algorithm 1.

5 Computational experiments

To demonstrate the effectiveness of the proposed Algorithm 1, some experiments were conducted with 12 crisp instances. Each one has items with random weights following a Uniform Distribution in $[20,120]$, three bin types ($Z = 3$) with $W_j \in \{50,100,150\}$. Three different forms to correlate the capacity of each bin to its cost were used: $Polinomial : C_j = 15 * W_j + 32$, $Concave : C_j = 10 * \sqrt{W_j}$ and $Convex : C_j = 0.1 * W_j^{3/2}$. For each cost function it was generated a crisp VCSBPP instance with 25, 50, 75 or 100 items, thus producing 12 crisp instances.

These instances were generated using the tool described in [12] and they are available in `http://www.cimab.transnet.cu/files/InstancesPartialPacking.zip` on five formats (AMPL, EXCEL, LP, MPS, ZIMPL and the corresponding results in a XLS file).

Table 2 presents the results obtained by CPLEX for the 12 crisp instances. In some cases where the optimal solution is not obtained after one hour, the best solution with the corresponding gap is presented. The structure of the solution obtained for these 12 crisp instances is presented in terms of the number of used bin of each type ($B[1], B[2], B[3]$) corresponding to $W_j = 50, 100, 1500$, respectively.

The columns Solutions and Time of Table 2 are referred to Algorithm 1. They show the number of solutions generated (it corresponds to $(B[1]+1) * (B[2]+1)(B[3]+1)$, as it was previously explained) and the execution time (in ms) for each case. The last column indicates the number of different types that are used in the optimal solution of each crisp instance. It is worth noting that all types are only used in the optimal solution of 1 crisp instance out of 12. Experiments were performed in an Intel Core i7 (without parallelization), 16 Gb of RAM and Zorin OS (Linux Ubuntu 18.04) operating systems. A value of 0 in column Time indicates a value that is smaller than 1 ms.

Instance	Solution	Gap	$B[1]$	$B[2]$	$B[3]$	Solutions	Time	Used types
N25Li	28916.00	0.00	0	1	12	26	0	2
N25Cc	1569.69	0.00	0	1	12	26	0	2
N25Cv	2304.54	0.00	0	1	12	26	0	2
N50Li	57082.00	0.00	0	3	23	96	2	2
N50Cc	3116.91	0.00	0	3	23	96	2	2
N50Cv	4406.81	1.11	0	11	18	228	51	2
N75Li	83684.00	3.08	0	1	36	74	3	2
N75Cc	4509.08	0.50	0	1	36	74	3	2
N75Cv	6511.35	0.25	0	10	30	341	15	2
N100Li	110318.00	1.32	0	2	47	144	9	2
N100Cc	5949.49	1.19	1	0	48	98	6	2
N100Cv	8616.40	0.95	1	16	38	1326	149	3

Table 2: Results of the solver for the crisp instances

In order to make more explicit the results on each crisp instances we present a series of graphics (Figures 5, 6 and 7), grouped by the type of cost function, where the y-axis shows the fitness function value (cost) and the x-axis presents their corresponding values of membership ($R[s]$, similar to Figure 4). To enhance the comprehension of these results, only the values of membership above 0.8 are presented. This may be considered as a preliminary value of $\alpha = 0.8$ in terms of α-cuts. Based on this value of α, Figure 5 shows five solutions of the 0.8-cuts for the instance with 25 items (up,left). The best solution for this case is the one with cost 24352 and feasibility of 0.841. In the same figure, if $\alpha = 0.9$ then the 0.9-cut only contains 3 solutions and the best cost is associated with the one with cost 26634 and feasibility of 0.922. Similar analysis may be conducted in Figures 6 and 7. It is interesting to note the difference of the distribution of the solutions of the instances with convex cost (see Figure 7).

In the case of Figure 7 with 50 items (up,right), 75 items (down,left) and 100 items (down,right), there are some points represented as white circles. These solutions are to be ignored by the decision maker because they are dominated by other solutions that are better both in terms of feasibility and cost. Thus, there is no rationale to select these solutions. They are included in the graph to illustrate the necessity to take this situation into account.

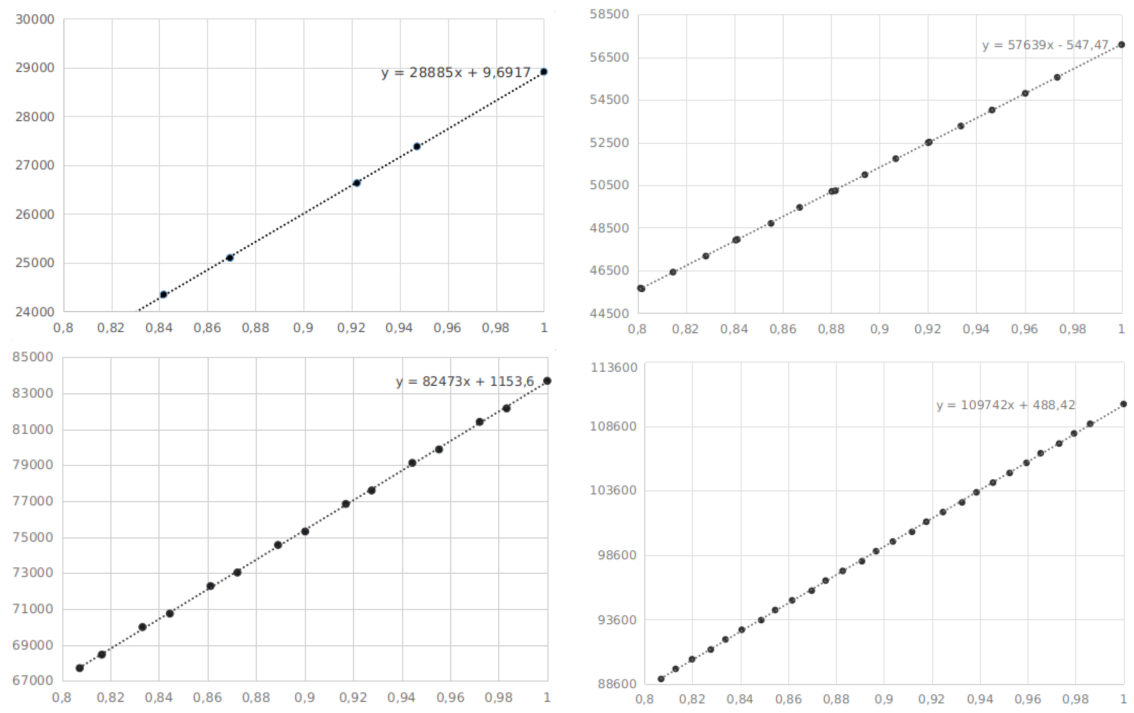

Figure 5: Fuzzy solutions for the crisp instances with Linear cost: 25 items (up,left), 50 items (up,right), 75 items (down,left) and 100 items (down,right)

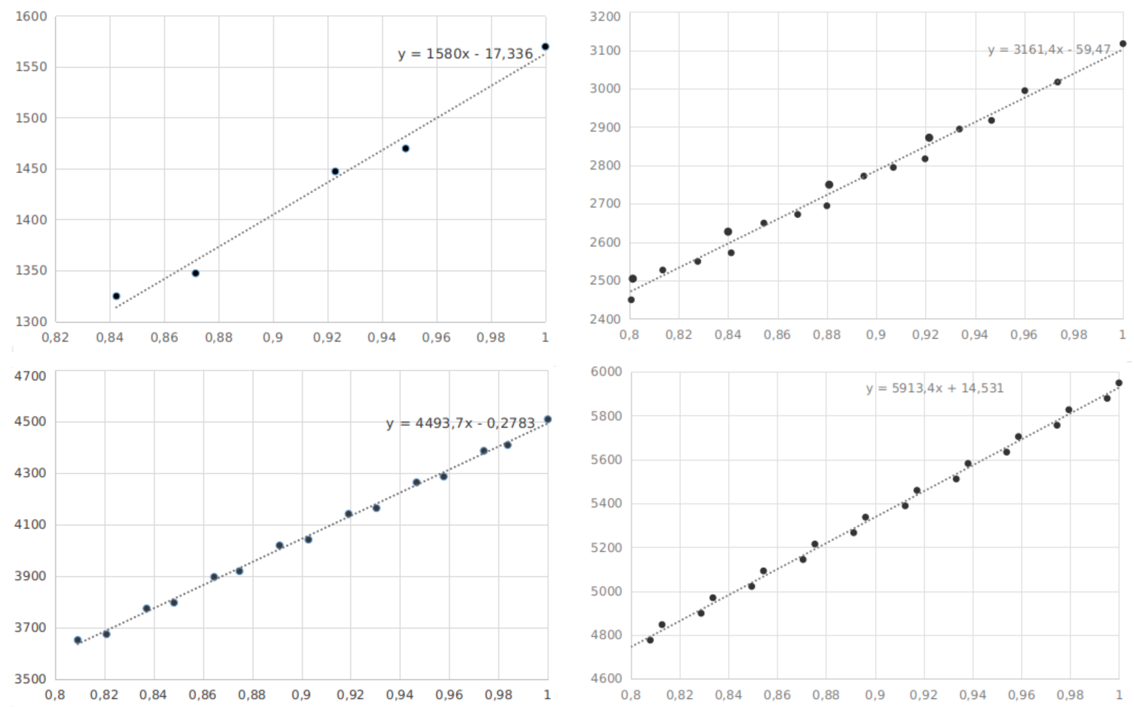

Figure 6: Fuzzy solutions for the crisp instances with concave cost: 25 items (up,left), 50 items (up,right), 75 items (down,left) and 100 items (down,right)

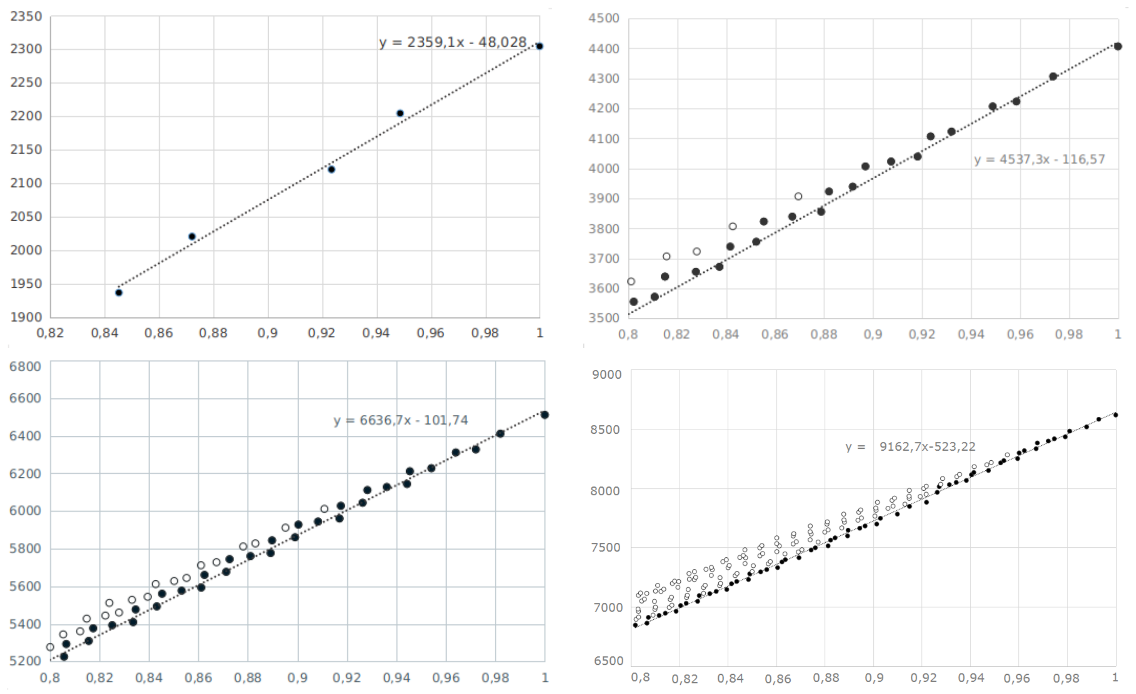

Figure 7: Fuzzy solutions for the crisp instances with convex cost: 25 items (up,left), 50 items (up,right), 75 items (down,left) and 100 items (down,right)

With respect to the rectification of the membership (associated to the function *RectifyMembership* in Algorithm 1) it was only useful in two instances: N50Cv (convex with 50 items) and N100Cv (convex with 100 items). To have a precise idea of what is happening after applying the *RectifyMembership* function, we present some examples of the improvement of membership(from $M[s]$ to $R[s]$) for certain solutions in Table 3, representing the greatest, the median and the smallest improvements. It is worth noting in Table 3 that the first solution improves its membership from $M[s] = 0.799$ to $R[s] = 0.808$, thus it would be ignored with $\alpha = 0.8$ if the rectification is not introduced.

Instance	$Q[s]$	$M[s]$	$R[s]$	Repacked w
N50Cv	3488.25	0.799	0.808	33.22
N50Cv	3306.81	0.711	0.718	27.70
N50Cv	3104.54	0.708	0.714	22.15
N100Cv	7881.55	0.918	0.922	31.66
N100Cv	7416.40	0.842	0.845	23.86
N100Cv	2891.03	0.329	0.332	20.25

Table 3: Examples of the improvements produced by the function *RectifyMembership*

In spite of these examples, in general terms the importance of the rectification was not very relevant in our experiments, maybe due to the inner solution strategy of CPLEX. It seems that it is more biased to produce less weighted bins, that are well used by Algorithm 1 to be considered the first to be freed. Further experiments with other ways of obtaining the initial solution (other solvers or by using heuristics algorithms such as FFD) may reveal the importance of the application of the *RectifyMembership* function.

A general picture of the improvement in the membership is presented in Table 4 in terms of the extremes values (min and max) and the median of the repacked weight in the two mentioned instances. Column *Solutions* presents the number of sampled instances (same as in Table 2), *Rectified* presented the number of solutions that were improved, *Max*, *Min* and *Ave* correspond to the maximum, the minimal and the average weight repacked. As the rectification derives in solutions with similar costs and improved memberships, they are preferable.

Instance	Solutions	Rectified	Max.	Min.	Ave.
N50Cv	228	41	33.22	22.15	23.58
N100Cv	1326	606	31.66	20.25	24.06

Table 4: Global view of the improvements produced by *RectifyMembership*

5.1 Scalability of the algorithm

Previous experiments were focused on the amount of items, and it was shown that this aspect increases the complexity of the instances. But, for Algorithm 1 the time complexity comes from the amount of bin types included in the solution given by the solver. The worst case occurs when all available types are used, and when there are a similar number of bins of each type. In Table 2 it is observed that all types are only used in 1 out of 12 cases. In addition, only 6 out of 12 solutions use more than 1 bin of the second more used type. It may be observed that the number of solutions (and the execution time) is directly related to this situation.

In the extreme case where the optimal solution consists of $|I|$ bins of different types ($Z = |I|$) to pack $|I|$ items, the number of solutions to be explored by Algorithm 1 is $\prod_{z=1..Z}[B[z]+1] = \prod_{z=1..|I|}[1+1] = 2^{|I|}$ according to expression (18). This implies that in the worst case the Algorithm 1 derives to the explicit enumeration of all subsets of I, thus its time complexity is $O(2^{|I|})$ in the worst case. Based on the results presented in Table 2, this seems to be unlikely. Indeed, this situation is not in the line of the nature of packing, i.e. placing several items together in a bin. We now extend the experimental study to explore the scalability of the Algorithm 1.

In this sense, six new instances were generated and solved in the same way than the previous ones, they have 100 items and the same three cost functions. The differences are in the weight range of the items and the number of bin types. The range was modified to assure that every bin type is a candidate to be used.

- 6 bin types, items weights randomly drawn from a uniform distribution in $[20, 160]$
- 12 bin types, items weights randomly drawn from a uniform distribution in $[10, 280]$

In Table 5 these six new instances are presented (along with the solution obtained by using CPLEX), in addition to the instances with 100 items and 3 types shown in Table 2. The number of available bin types in each instance is included in the identifier used for each instance, e.g. N100Li3 identifies the instances with linear cost and 3 bin types. The last column shows the number of different types of bins that are actually used in the optimal solution. These larger instances are available at `http://www.cimab.transnet.cu/files/LargeInstancesFVCSBPP.zip`

In Table 6 there are 12 columns related to each bin type: $W_1 = 25$, $W_2 = 50$, $W_3 = 75$, $W_4 = 100$, $W_5 = 125$, $W_6 = 150$, $W_7 = 175$, $W_8 = 200$, $W_9 = 225$, $W_{10} = 250$, $W_{11} = 275$, $W_{12} = 300$. Each type that is not available in an instance is reflected as $-$.

It must be remarked that the last instance, N100Cv12 was not fully solved due to the high amount of time needed to generate all the possible crisp solutions, calculated in 4342947840. It is important to note that this large number of solutions is only $3 * 10^{-21}$ % of the entire space of combinations ($2^{100} > 10^{30}$). Table 5 only reflects the results within half an hour of execution, having in this time a membership value of 0.85. It is easy to note that the generation of all solutions in two instances with convex cost function (N100Cv6, N100Cv12) would take a considerably time.

Instance	Solution	Gap	Solutions	Time	Used types
N100Li3	110318.00	1.32	144	9	2
N100Cc3	5949.49	1.19	98	6	2
N100Cv3	8616.40	0.95	1326	149	3
N100Li6	144324.00	1.39	2520	162	3
N100Cc6	7297.58	1.16	1176	64	3
N100Cv6	11252.29	0.75	11104128	852638	6
N100Li12	224007.00	1.02	1716	229	3
N100Cc12	8693.89	2.43	840	323	3
N100Cv12	21735.06	0.12	28630410	1800000	10

Table 5: Results for larger instances

Instance	$B[1]$	$B[2]$	$B[3]$	$B[4]$	$B[5]$	$B[6]$	$B[7]$	$B[8]$	$B[9]$	$B[10]$	$B[11]$	$B[12]$
N100Li3	-	0	-	2	-	47	-	-	-	-	-	-
N100Cc3	-	1	-	0	-	48	-	-	-	-	-	-
N100Cv3	-	1	-	16	-	38	-	-	-	-	-	-
N100Li6	-	0	0	0	6	7	44	-	-	-	-	-
N100Cc6	-	0	0	0	3	5	48	-	-	-	-	-
N100Cv6	-	7	20	17	23	16	8	-	-	-	-	-
N100Li12	0	0	0	0	0	0	0	0	3	0	10	38
N100Cc12	0	0	0	1	0	0	0	0	0	0	9	41
N100Cv12	0	0	11	16	11	6	10	7	14	3	11	3

Table 6: Number of bins of each type used in the optimal solution

In addition, it makes no sense to present such a considerable number of solutions to a decision maker. Indeed, in future works we think that it is important to study alternative ways to reduce the number of interesting solution to be presented. Obvious options are to focus on certain intervals of membership and cost that may be inserted in the general structure of Algorithm 1. Another trivial extension is to introduce a random sampling instead of sequential enumeration in function *next* described in Algorithm 2. However, other ideas could be explored, such as interactively exploring the most interesting portions of the spectrum of solutions, guided by the user preferences.

In general, it may be observed that only in the two largest instances with convex cost, more than three types of bins are used in the optimal solution; fact that increase, considerably, the runtime. The others (including the two instances with 100 items, 12 bin types, with linear and concave cost) are solved in less than a second.

6 Discussion

Firstly, Table 1 shows that the number of solutions for each fuzzy problem is significantly lower than the ones that may be expected if all possible subsets $\tilde{I}$ are considered. For example, in the solution obtained by CPLEX for the crisp instance N25Li (linear cost and 25 items) where no bins of 50 were used, 1 bin of 100 and 12 bins of 150, then $(0+1)*(1+1)*(12+1) = 26$ solutions are produced by the Algorithm 1 for the fuzzy problem, including the non-relaxed one.

Notice that for the biggest instances (e.g, N100Li, the one with 100 items and linear cost), the number of samples is also very low $(0+1)*(2+1)*(47+1) = 144$ which demonstrate the efficiency of the proposed method. For the first case, 26 sampled instances is more than one million times smaller than $2^{25} = 33554432$ while for the one with 100 items (2^{100}) is $1.26*10^{30}$ bigger than 144.

The graphics presented are also an important tool for the decision maker in order to select the level of relaxation that fits better her/his requirements. Indeed, the tendency lines presented in each figure illustrate the general trade-off. For example, in Figure 7 it can be found a tendency line for the set of non-dominated solutions for the instance with 100 items, with the form $y = 9162,7x - 523,22$ where the y-axis represents the cost and the x-axis indicates the membership value. In these figures, the solutions below the tendency are likely to be preferred because they reach a proportion of the cost with respect to the relaxation (membership) that is better than the average.

Following with the same instance, it is interesting to note that the tendency lines allow another analysis, in addition to the the crisp alternative (in this instance, the exact solution with full packing has a $cost \cong 8616,40$. Indeed, the function is useful to know which level of relaxation is needed to obtain a partial packing with cost under 7000. Based on the tendency $7000 = 9162,7x - 523,22$, we can find that $x = (7000+523,22)/9162,7 = 0,821$, i.e., with a solution with cost of 7000 it is possible with a partial packing with membership of 0.821. That is, a decision maker may obtain a solution within an available budget, based on the corresponding graphic and the tendency lines thus making a more informed decision.

It is also interesting to note the impact that the cost/capacity ratio has on the number of solutions generated. Firstly, it must be assessed in Table 2 that the number of solution for the instance with 100 items and convex cost function is considerably higher than the one with 100 items and linear and concave cost functions. Except for the instances with 25 items, in all the other sizes this phenomenon can be observed too. It is also interesting to note that this phenomenon is also correlated with the appearance of the dominated solutions (white dots). This aspect is clearly observed in Figure 7, particularly in the instances with 75 and 100 items.

It is noticeable the effect of the convex cost function because smaller bins are preferable in this case, i.e. it is cheaper to use three bins with size equals to 50 (cost=35.36, thus overall cost of $35.36 * 3 = 106.08$) instead of using one bin with size equals to 150 (cost=183.71). In general, the instances that have more dominated solution are the ones that uses the higher number of bins with smaller capacities (50 and 100) in their crisp solutions (see Table 2). This complexity of the fuzzy solution in terms of the number of interesting solutions (that depends on the nature of each instance) becomes even more important to use an efficient alternative of the traditional parametric approach, as the one presented in this paper.

Regarding this topic, the results shown in Table 5 and 6 support the fact that the convex cost function increases the complexity of the solution of the fuzzy problem (as it was previously shown in [12] for crisp instances of VCSBPP). This type of function tends to produce a large number of bin types in the structure of the solution given by the solver. The algorithm efficiency decreases as the number of bin types included increases. Consequently, this poses a problem in order to present a small and meaningful set of solutions to the decision maker. In spite of this aspect that require further studies, the performance of the heuristic Algorithm 1 is adequate even for the largest instances, in comparison with the whole exploration of alternatives. Thus, it seems to be an effective way to solve fuzzy instances in a relative small time.

In general, the experimental results show that it is possible to obtain very quickly an interesting set of solutions (i.e. a family of crisp solutions that is a fuzzy solution of a fuzzy problem) for the VCSBPP that allow a more informed decision making.

7 Conclusions

This paper presents a fuzzy version of the Variable Cost and Size Bin Packing Problem (VCSBPP), that allow partial packing, i.e. some items are not packed. This fuzzy VCSBPP seems to be useful to tackle many real-life situations where the budget is not sufficient to pack all items.

The proposed fuzzy version of VCSBPP is expressed in terms of a fuzzy satisfaction of the condition related to the complete packing, by using a fuzzy definition of the universal quantifier. Based on the parametric approach, the solution of the fuzzy problem consists of a set of solutions that shows different trade-offs between relaxation (violation of the original condition) and benefit (cost reduction).

The problem of obtaining good relaxations of the fuzzy problem is solved by a new efficient heuristic algorithm that obtains interesting relaxed solutions (with partial packing). The experiments show that the trade-off between cost and satisfaction of the original packing is more complex when the number of bin types included in the crisp solution increases, particularly when convex cost function is used.

With this tool a decision maker can better fulfill the requirements of real-world applications of VCSBPP, allowing partial packing but maintaining the highest possible degree of membership to the original conditions.

References

[1] Jørgen Bang-Jensen and Rune Larsen. Efficient algorithms for real-life instances of the variable size bin packing problem. *Computers and Operations Research*, 39(11):2848–2857, 2012.

[2] Fatih Emre Boran, Diyar Akay, and Ronald R Yager. An overview of methods for linguistic summarization with fuzzy sets. *Expert Systems With Applications*, (61):356377, 2016.

[3] Isabel Correia, Luis Gouveia, and Francisco Saldanha-da Gama. Solving the variable size bin packing problem with discretized formulations. *Computers & Operations Research*, 35(6):2103–2113, 2008.

[4] Teodor Gabriel Crainic, Luca Gobbato, Guido Perboli, Walter Rei, Jean-Paul Watson, and David L Woodruff. Bin Packing Problems with Uncertainty on Item Characteristics: An Application to Capacity Planning in Logistics. *Procedia - Social and Behavioral Sciences*, 111:654–662, 2014.

[5] Teodor Gabriel Crainic, Guido Perboli, Walter Rei, and Roberto Tadei. Efficient lower bounds and heuristics for the variable cost and size bin packing problem. *Computers & Operations Research*, 38(11):1474–1482, 2011.

[6] Arup Dey and Kais Zaman. A robust optimization approach for solving two-person games under interval uncertainty. *Computers & Operations Research*, 119:104937, 2020.

[7] György Dósa and Leah Epstein. The tight asymptotic approximation ratio of First Fit for bin packing with cardinality constraints. *Journal of Computer and System Sciences*, 96:33–49, 2018.

[8] Ali Ebrahimnejad and Jose Luis Verdegay. *Fuzzy Sets-Based Methods and Techniques for Modern Analytics*. Studies in Fuzziness and Soft Computing. Springer International Publishing, 1 edition, 2018.

[9] Ugur Eliiyi and Efendi Nasibov. A fuzzy perspective for two-dimensional packing of variable sized items. In R Kasmbeyli, S Özpeynirci C. Dinçer, and L Sakalauskas, editors, *24th Mini EURO Conference Continuous Optimization and Information-Based Technologies in the Financial Sector*, pages 177–182. Vilnius, 2010.

[10] Virgilio C Guzmán, David A Pelta, and José L Verdegay. An approach for solving maximal covering location problems with fuzzy constraints. *International Journal of Computational Intelligence Systems*, 9(4):734–744, 2016.

[11] Vera Hemmelmayr, Verena Schmid, and Christian Blum. Variable neighbourhood search for the variable sized bin packing problem. *Computers & Operations Research*, 39:1097–1108, 2012.

[12] Jorge Herrera-Franklin, Alejandro Rosete, Milton García-Borroto, and Suitberto Cabrera-García. Influencia de distribuciones estadísticas en la complejidad de instancias del problema de empaquetamiento con tamaño y costo variable. *Revista de Ingeniería Mecánica*, 23(2), 2020.

[13] Jorge Herrera-Franklin, Alejandro Rosete, Milton García-Borroto, Carlos Cruz-Corona, and David A Pelta. On the Impact of Fuzzy Constraints in the Variable Size and Cost Bin Packing Problem. In Marie-Jeanne Lesot, Susana Vieira, Marek Z Reformat, João Paulo Carvalho, Anna Wilbik, Bernadette Bouchon-Meunier, and Ronald R Yager, editors, *Information Processing and Management of Uncertainty in Knowledge-Based Systems*, pages 230–240, Cham, 2020. Springer International Publishing.

[14] Nguyen Van Hung, Vo Tam, Nguyen Tuan, and Donal O'Regan. Convergence analysis of solution sets for fuzzy optimization problems. *Journal of Computational and Applied Mathematics*, 369:112615, 2020.

[15] Akshay Jain, Mihail Popescu, James Keller, Marilyn Rantz, and Brianna Markway. Linguistic summarization of in-home sensor data. *Journal of Biomedical Informatics*, 96:103240, 2019.

[16] Kshitija Kalaskar. Variable Size Bin Packing Algorithm for IoT. *International Journal of Science and Research (IJSR)*, 4(7):2599–2602, 2013.

[17] Jangha Kang and Sungsoo Park. Algorithms for the variable sized bin packing problem. *European Journal of Operational Research*, (147):365–372, 2003.

[18] Jong-Kyou Kim, H Lee-Kwang, and Seung W. Yoo. Fuzzy bin packing problem. *Fuzzy Sets and Systems*, 120(3):429–434, 2001.

[19] Zeng Meng, Yongsheng Pang, Yuxue Pu, and Xuan Wang. New hybrid reliability-based topology optimization method combining fuzzy and probabilistic models for handling epistemic and aleatory uncertainties. *Computer Methods in Applied Mechanics and Engineering*, 363, 2020.

[20] Amin Salih Mohammed, Saravana Balaji, Saleem Basha, Asha, and Venkatachalam. FCO Fuzzy constraints applied Cluster Optimization technique for Wireless AdHoc Networks. *Computer Communications*, 154:501–508, 2020.

[21] Ali Mortazavi. A new fuzzy strategy for size and topology optimization of truss structures. *Applied Soft Computing*, 93:106412, 2020.

[22] Mario Piattini, José Galindo, and Angélica Urrutia. *Fuzzy Databases: Modeling, Design and Implementation*. 2006.

[23] Seyed Mahdi Shavarani, Sam Mosallaeipour, Mahmoud Golabi, and Gokhan Izbirak. A congested capacitated multi-level fuzzy facility location problem: An efficient drone delivery system. *Computers and Operations Research*, (108):57–68, 2019.

[24] M Torres, D A Pelta, and M Teresa Lamata. A New Approach for Solving Personalized Routing Problems with Fuzzy Constraints. In *2018 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, pages 1–5, 2018.

[25] Camino Vela, Sezin Afar, Juan Palacios, Ines Gonzalez Rodriguez, and Jorge Puente Peinador. Evolutionary tabu search for flexible due-date satisfaction in fuzzy job shop scheduling. *Computers & Operations Research*, 119:104931, 2020.

[26] J L Verdegay. Fuzzy mathematical programming. *Fuzzy information and decision processes*, pages 231–237, 1982.

[27] Shanshan Wang, Jinlin Li, and Sanjay Mehrotra. Chance-Constrained Bin Packing Problem with an Application to Operating Room Planning. 2019.

[28] R. Yager, J. Kacprzyk, and S. Zadrożny. A Fuzzy Logic Based Approach to Linguistic Summaries of Databases. *International Journal of Applied Mathematics and Computer Science*, Vol. 10(no 4):813–834, 2000.

[29] Ronald R Yager. A new approach to the summarization of data. *Information Sciences*, 1(28):6986, 1982.

[30] Ronald R. Yager. Database discovery using fuzzy sets. *International Journal of Intelligent Systems*, 11(9):691–712, 1996.

[31] Shuai Zhang, Mingzhou Chen, Wenyu Zhang, and Xiaoyu Zhuang. Fuzzy optimization model for electric vehicle routing problem with time windows and recharging stations. *Expert Systems with Applications*, 145:113123, 2019.