Journal of Information Systems Engineering & Management
2019, 4(4), em0104
ISSN: 2468-4376

A Percentile Methodology Applied to Binarization of Swarm Intelligence Metaheuristics

Matias Valenzuela [1]*, Hernan Pinto [1], Paola Moraga [1], Francisco Altimiras [1], Gabriel Villavicencio [1]

[1] *Pontificia Universidad Católica de Valparaíso,* Valparaíso, CHILE

***Corresponding Author:** matias.valenzuela@pucv.cl

Citation: Valenzuela, M., Pinto, H., Moraga, P., Altimiras, F. and Villavicencio, G. (2019). A Percentile Methodology Applied to Binarization of Swarm Intelligence Metaheuristics. *Journal of Information Systems Engineering & Management*, 4(4), em0104. https://doi.org/10.29333/jisem/6348

Published: December 7, 2019

ABSTRACT

The binarization mechanisms of continuous metaheuristics are of interest in operational research. This is mainly due to the fact that there are a lot of combinatorial problems that are NP-hard. In this article, we exploit the concept of percentile as a mechanism of binarization of swarm intelligence continuous metaheuristics. To evaluate the behavior of our binary operator, the Multi-verse metaheuristic is used and applied to solve the combinatorial problem of the knapsack. The binary algorithm obtained, the binary multi-verse Optimizer (BMVO) shows good performance in solving the most difficult problems of the knapsack.

Keywords: metaheuristics, multidimensional knapsack problem, binarization, percentile

INTRODUCTION

In the industry, optimization problems are relevant, particularly at the level of decision making. Behind a decision-making process, there is always an optimization problem. Many of these problems have their domain in binary spaces. For example, we can find binary space problems in Civil Engineering, Bio-Informatics (Barman and Kwon, 2017), Operational Research (Crawford et al., 2017, September; 2018; Garcia and Măntoiu, 2014; García et al., 2019a), resource allocation (Astorga et al., 2018; García et al., 2017, September, 2019b; Valenzuela et al., 2019, June) scheduling problems (García et al., 2017, February, 2018a), routing problems among others.

Another line of research that has had an important impact is the design of algorithms inspired by natural phenomena to solve optimization problems. Many natural phenomena are developed in continuous spaces, so it is common to find a large number of these algorithms that work properly continuously, As examples of these algorithms we have Cuckoo Search, Black Hole, Bat Algorithm, and Multi-verse Optimization Algorithm (Mirjalili et al., 2016) among others. It is a challenge to transform a continuous algorithm into its binary version without altering the exploration and exploitation processes characteristic of each metaheuristic and that subsequently it adequately performs in combinatorial problems.

When a state of the art is realized, there are several techniques that give a solution to the complexity of passing an algorithm designed for continuous spaces in an algorithm that solves binary problems. We have general and specific techniques. However, these techniques work well for some problems and not for others. In general, the main techniques are transfer functions, angle modulation and the design of quantum operators. If you want to deepen this line, we recommend reading (Crawford et al., 2017). The objective of this article is to use the binarization technique that uses the percentile method and performs a binarization process. The multiverse optimizer (MVO) technique was chosen because it has been used in several continuous problems but in few combinatorial problems. MVO has been applied to voltage stability and voltage deviation problems in Trivedi et al. (2016, March). In Kumar and Suhag (2017), they applied MVO to control multi source hydrothermal power system.

Three concepts use the MVO algorithm, the white hole, the black hole who performs the exploration in the search space and wormholes that are in charge of performing the exploitation. The analogy says that each solution corresponds to a universe and associates a parameter called inflation rate which is proportional to the value of its fitness function. The black and white holes allow to exchange of objects of the universes and the probability that an exchange is made is handled by the inflation rate indicator. The solutions are ordered at the end of their inflation rate after each solution generates a white hole and the solution that is sent through the white hole is selected randomly. The replacement of the different objects is regulated by Equation 1. According to this equation, the lower the inflation rate, the greater the probability that the objects of that solution are replaced. However, there is only an exchange between universes, but there is no concept of mutation or disturbance of a universe.

$$x_j = \begin{cases} x_k^j \; r_1 < NI(U_i) \\ x_i^j \; r_1 > NI(U_i) \end{cases} \tag{1}$$

where x_i^j indicates the jth parameter of the ith solution, U_i corresponds to ith solution, $NI(U_i)$ is normalized inflation rate of the ith solution, r_1 is a random number in [0,1], and x_k^j indicates the jth parameter of kth solution selected by roulette wheel selection.

To mutate the objects, the concept of wormhole is used. The whormhole is established between each of the solutions and the best solution. The exchange of objects is produced from the best solution and a perturbation of the original object is made. The mechanism that regulates this perturbation is shown in Equation 2.

$$x_i^j = \begin{cases} \begin{cases} x_j + TDR_x\left(\left(ub_j - lb_j\right)xr_4 + lb_j\right) \; r_3 < 0.5 \\ x_j - TDR_x\left(\left(ub_j - lb_j\right)xr_4 + lb_j\right) \; r_3 \geq 0.5 \end{cases} & r_2 < WEP \\ x_i^j & r_2 \geq WEP \end{cases} \tag{2}$$

In this expression WPE and TDR correspond to coefficients defined in Equations 3 and 4 respectively. *lbj* corresponds to the lower bound and *ubj* to the upper bound of the jth variable. *xj* represents to the variable of the jth dimension of the i solution. The numbers r2, r3 and r4 correspond to a random numbers between [0,1].

$$WEP = min + lx\left(\frac{max - min}{L}\right) \tag{3}$$

$$TDR = 1 - \left(\frac{l^{\frac{1}{p}}}{L^{\frac{1}{p}}}\right) \tag{4}$$

To check the binary multi-verse optimizer algorithm (BMVO) using percentile technique, we use the well-known knapsack problem. Experiments were developed using a random operator to validate the contribution of the percentile technique in the binarization pro-cess of the multi-verse optimizer algorithm. In addition, local search operator is used to strengthen the results. Moreover, the binary artificial algae (BAAA) and K-means transition ranking (KMTR) algorithms were used to compare our results. (BAAA) was developed in Zhang et al. (2016) and uses transfer functions to perform the binarization process. KMTR was developed in García et al. (2018b) and uses a K-means algorithm to perform the binarization. The results show that the percentile technique obtains results superior to those obtained by the random operator and that our BMVO algorithm shows competitive results against the BAAA and KMTR algorithms.

KnapSack PROBLEM

One of the NP-hard problems that has been studied most, corresponds to the Knapsack problem considering the large number of variants that it has. In this article, we will use the multidimensional knapsack problem to test our percentile binarization. The problem of the multidimensional knapsack (MKP) focuses on resource allocation problems. The objective is to find a subset of objects that produce greater benefit satisfying the different restrictions of the problem. Despite a large number of studies, this problem remains a challenge due to the difficulties in solving medium and large-sized instances. Performing a small state of the art of MKP, we find that this has been addressed in Haddar et al. (2016) using a quantum binarization technique, in García et al. (2018b) they applied the technique of grouping k-means to perform binarization, an improved optimization of The fruit fly in Meng and Pan (2017), in Liu et al. (2016) a differential algorithm with transfer functions was used, and in Bansal and Deep (2012) a modification of the PSO equations was used. MKP can be configured as:

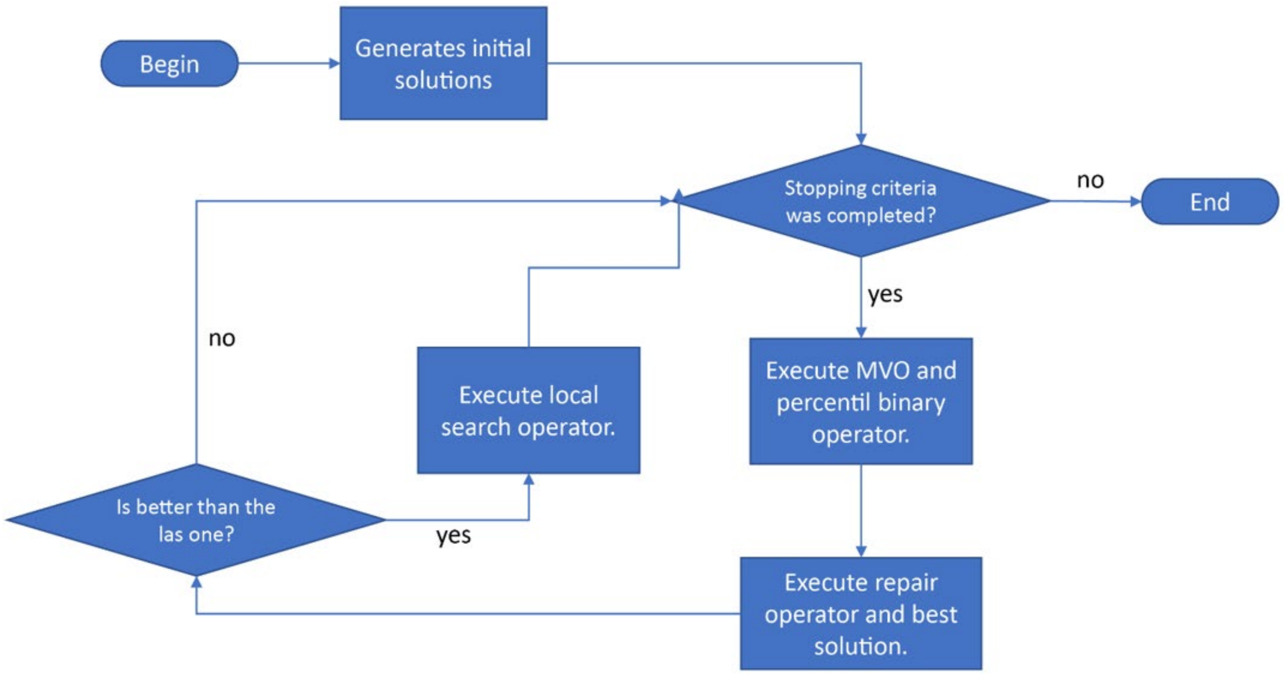
Begin
Generates initial solutions
Stopping criteria was completed?
no
End
yes
no
Execute local search operator.
Execute MVO and percentil binary operator.
Is better than the las one?
yes
Execute repair operator and best solution.

Figure 1. Flowchart of the binary Multi-verse optimizer algorithm

$$\text{Maximize} \sum_{j=1}^{n} p_j x_j \tag{5}$$

$$\text{subject to} \sum c_{ij} x_{ij} \leq b_i, i \in \{1, \dots, m\} \tag{6}$$

with $x_j \in \{1, \dots, m\}$. p_j corresponds to the profit of element j. c_{ij} represents a cost associated with dimension i and element j. The constraints in each dimension i are represented by b_i. The solution can be modelled using a binary representation, in this representation a 0 means the element is not included in the knapsack.

BINARY MULTI-VERSE OPTIMIZER ALGORITHM

Due to the high computational complexity of MKP, the binary multi-verse optimization algorithm (BMVO) is composed of 4 operators to successfully solve the MKP. The operators corresponding to an initialization operator, a local search operator, the binary operator that uses the percentile technique and a repair operator in the case of solutions that do not meet any of the restrictions. The general flow chart of the algorithm is shown in Figure 1. The first stage corresponds to the initiation of solutions which will be detailed in section III-A. Subsequently, it is verified if the algorithm meets the detention conditions which is associated with a maximum number of iterations. In case of not complying with them, the MVO algorithm is executed in its original form and the solutions obtained by MVO are binarized by the percentile operator, this part of the algorithm is detailed in III-B. After having performed the binarization, we must verify the consistency of the solutions with their constraints and the results obtained are compared against the best solution generated so far. If the new solution is superior, it is replaced and a local search operator is executed.

Initialization and Element Weighting

As BMVO is a swarm algorithm, to begin the exploration and exploitation of the search space, the list of solutions must be initialized. For the generation of each solution, an element is randomly chosen first. The next step is to check if other elements can be incorporated, for this we must evaluate the constraints of our problem. To select the new element, we generate a list of possible elements that comply with the constraints. For each element is calculated its weight and the element which has the best weight is selected. This procedure is repeated until no additional element can be incorporated. In Figure 2, the initialization algorithm is displayed.

To calculate the weight of each element, several methods have been developed. In Pirkul (1987) a pseudo-utility in the surrogate duality approach was proposed. The way to calculate it is shown in the equation 7. In this equation the variable w_j corresponds to the surrogate multiplier whose value is between 0 and 1. This multiplier can be interpreted as a shadow prices of the jth constraint.

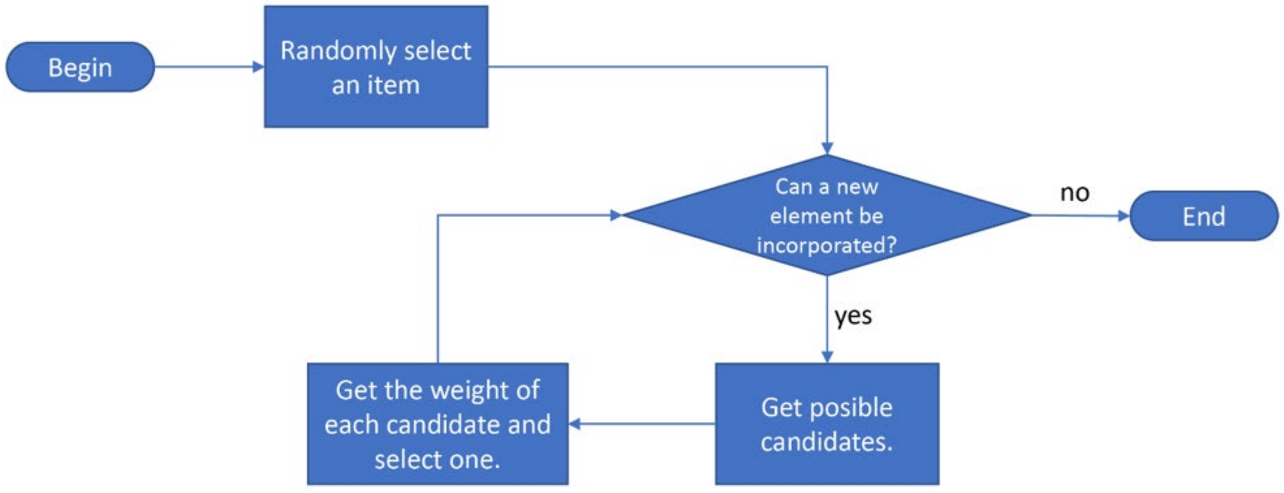
Begin
Randomly select an item
Can a new element be incorporated?
no
End
yes
Get posible candidates.
Get the weight of each candidate and select one.

Figure 2. Flowchart of generation of a new solution

$$\delta_i = \frac{p_i}{\sum_{j=1}^{m}(w_j c_{ij})} \tag{7}$$

A more intuitive measure focused on the avergage resource occupancy was proposed by García et al. (2017, September). It is shown in equation 8.

$$\delta_i = \frac{\sum_{j=1}^{m}\left(\frac{c_{ij}}{m_{bj}}\right)}{p_i} \tag{8}$$

In this article we used a variation of this last measure focused on the average occupation and proposed in García et al. (2018b). this variation considers the elements that exist in knapsacks to calculate the average occupancy. In each iteration depending on the selected items in the solution the measure is calculated again. The expression of this new measure is shown in equation 9.

$$\delta_i = \frac{\sum_{j=1}^{m}\left(\frac{c_{ij}}{m\left(b_j - \sum_{i\in S}(c_{ij})\right)}\right)}{p_i} \tag{9}$$

Percentile Binary Operator

Due to the iterative nature of swarm intelligence algorithms and considering that MVO works in continuous space. The velocity and position of the solutions are updated in $\mathbb{R}^n$. a general way of writing the update is shown in the equation 10. In this equation $x_{t=1}$ represents the position of the particle x at time $t+1$. To obtain the position, we consider the function Δ, which is specific to each algorithm. For example in black hole $\Delta(x) = \alpha \oplus levy\ (\lambda)(x)$, and in PSO, Firefly, and Bat algorithm Δ can be written in simplified form as $\Delta(x)\ =\ v\ (x)$.

$$x_{t=1}\ =\ \Delta_{t+1}(x(t)) \tag{10}$$

For the case of MVO, it is considered a binary percentile operator to perform the passage of the continuous space to the binary space. Given the particle x, let us consider the magnitude of the displacement $Deltai(x)$ in the ith component and we group these magnitudes for all solutions in order to obtain the values for the percentiles 20, 40, 60, 80, 100. At each percentile, we will assign a transition probability where the values are shown in the Equation 11. Using these transition probabilities together with the Equation 12, binarization of the solutions is performed. The algorithm is detailed in Algorithm 1.

$$P_{tr}(x^i) = \begin{cases} 0.1 & \text{if } x^i \in \text{group}\{0,1\} \\ 0.5 & \text{if } x^i \in \text{group}\{2,3,4\} \end{cases} \tag{11}$$

$$x^i(t+1) = \begin{cases} x^i(t) & \text{if rand} < P_{tg}(x^2) \\ x^i(t) & \text{otherwise} \end{cases} \tag{12}$$

Algorithm 1. Percentile binary operator

```
1: **Function** percentilebinary (vList, pList)
2: **Input** vList, pList
3 **Output** pGroup Value
4: pValue = getPercentileValue (vList, pList)
5: **for each** value in vList do
6:          pGroup Value = getPercentileGroupValue(pValue, vList)
8: **end for**
9: **return** pGroupValue
```

Repair Operator

Local search and binary percentile operators can generate infeasible solutions. There are different mechanisms to address these infeasible solutions. In this article, the repair of the solutions was considered. To perform the repair, the measure described in the equation 9 was used. If the solution requires repair, the element with the maximum measure is chosen and it is removed from the solution, this process is iterated until a valid solution is obtained. The solution is then improved by exploring the possibility of incorporating new elements. This stage of improvement is iterated until there are no elements that can be incorporated without violating the constraints. The pseudocode is shown in Algorithm 2.

Algorithm 2. Repair Algorithm

```
1: Function Repair ($S_{in}$)
2: Input Input Solution $S_{in}$
3: Output The repair solution $S_{out}$
4: $S \leftarrow S_{in}$
5: While needRepair ($S$) = True do
6 :          $s_{max} \leftarrow$ getMaxWeight($S$)
7 :          $S_{max} \leftarrow$ removeElement ($S$, $s_{max}$)
8: end while
9: state ← False
10: while state == False do
11:          $s_{min} \leftarrow$ getMinWeight($S$)
12:          if $s_{min} == \emptyset$ then
13:                    state ← True
14:          else
15:                    $S \leftarrow$ addElement($S$, $s_{min}$)
16:          end if
17: end while
18: $S_{out} \leftarrow S$
19: return $S_{out}$
```

RESULTS

Insight of BMVO Algorithm

This section aims to find out the contribution of the per- centile binary operator to the performance of the algorithm. To carry out the comparison problems cb.5.250 from the OR-library were selected. Violin charts and the Wilcoxon non-parametric signed-rank test were used to perform the statistical analyzes. In the charts, the X axis identifies the studied instances and the Y axis the % -Gap described in the equation 13. The Wilcoxon test is run to determine if the results obtained by BMVO have significant difference with respect to other algorithms. The parameter settings and browser ranges are shown in **Table 1**.

$$\% - GAP = 100\ (\frac{Bestknown - SolutioValue}{Bestknown})$$

Table 1. Setting of parameters for binary multi-verse search algorithm

Parameters	Description	Value	Range
N	Number of solutions	30	[20, 25, 30]
G	Number of percentiles	5	[4,5,6]
Iteration Number	Maximum iterations	1000	[1000]
max	Maximum for WEP coefficient	1	1
min	Minimum for WEP coefficient	0.2	0.2
p	p parameter for TDR coefficient	6	6

Table 2. Evaluation of percentile binary operator

Set	Best Known	best *random.ls*	best BMVO	best *random.wls*	best *wls*	avg *random.ls*	avg BMVO	avg *random.wls*	avg *wls*
cb.5.250-0	59312	59211	59225	59158	59175	59132.1	59146.2	59071.8	59131.4
cb.5.250-1	61472	61435	61435	61409	61409	61324.6	61391.3	61288.3	61372.1
cb.5.250-2	62130	62036	62074	61969	61969	61894.4	61971.1	61801.6	61923.2
cb.5.250-3	59463	59367	59446	59365	59349	59257.8	59324.6	59136.1	59271.7
cb.5.250-4	58951	58914	58951	58883	58930	58725.6	58821.1	58693.6	58757.1
cb.5.250-5	60077	60015	60056	59990	60015	59904.6	59963.1	59837.8	59943.1
cb.5.250-6	60414	60355	60355	60348	60349	60208.2	60325.8	60230.6	60310.2
cb.5.250-7	61472	61436	61436	61407	61407	61290.8	61337.6	61233.9	61341.7
cb.5.250-8	61885	61829	61885	61790	61782	61737.1	61795.3	61644.9	61739.8
cb.5.250-9	58959	58832	58866	58822	58787	58769.1	58789.2	58653.7	58771.6
Average	60413.5	60343	60372.9	60314.1	60317.2	60224.4	60286.5	60159.2	60256.2
p-value							4.16 e-05		1.16 e-05

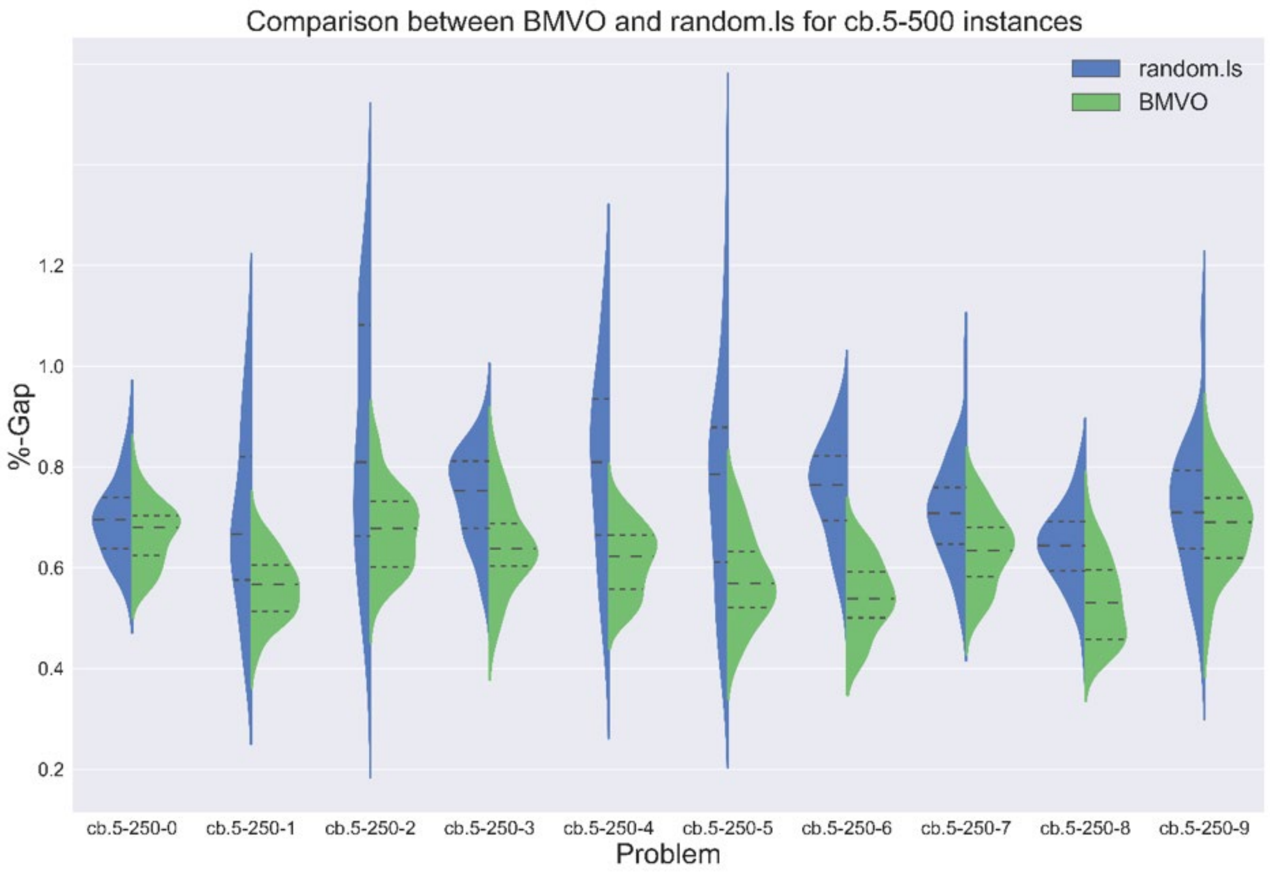

Figure 3. Evaluation of percentile binary operator with Local Search operator

1) Evaluation of percentile binary operator: A random operator was designed to evaluate the contribution of the percentile binary operator. This random operator has the pe- culiarity of performing the transitions with a fixed probability of 0.5 without considering in which percentile the variable is located. Two configurations were studied: The first one where the local search operator is included and the second where the local operator is not considered. BMVO corresponds to our standard algorithm. *random.ls* is the random variant that includes the local search operator. *BMVO.wls* corresponds to the version with percentile binary operator without local search operator. Finally, *random.wls* describes the random algorithm without local search operator.

When we compared the Best Values between BMVO and *random.ls* which are shown in Table 2. BMVO outperforms to *random.ls*. However, the Best Values between both algorithms are very close. In the Average comparison, BMVO outper- forms *random.ls* almost in all problems. The comparison of distributions is shown in Figure 3. We see the dispersion of the *random.ls* distributions are bigger than the dispersions of BMVO. In particular, this can be appreciated in the problems 1, 2, 4, 5, 6, and 9. Therefore, the percentile binary operator together with local search operators, contribute to the precision of the results. Finally, the BMVO distributions are closer to zero than *random.ls* distributions, indicating that BMVO has consistently better results than *random.ls*. When we evaluate the behaviour of the algorithms through the Wilcoxon test, this indicates that there is a significant difference between the two algorithms.

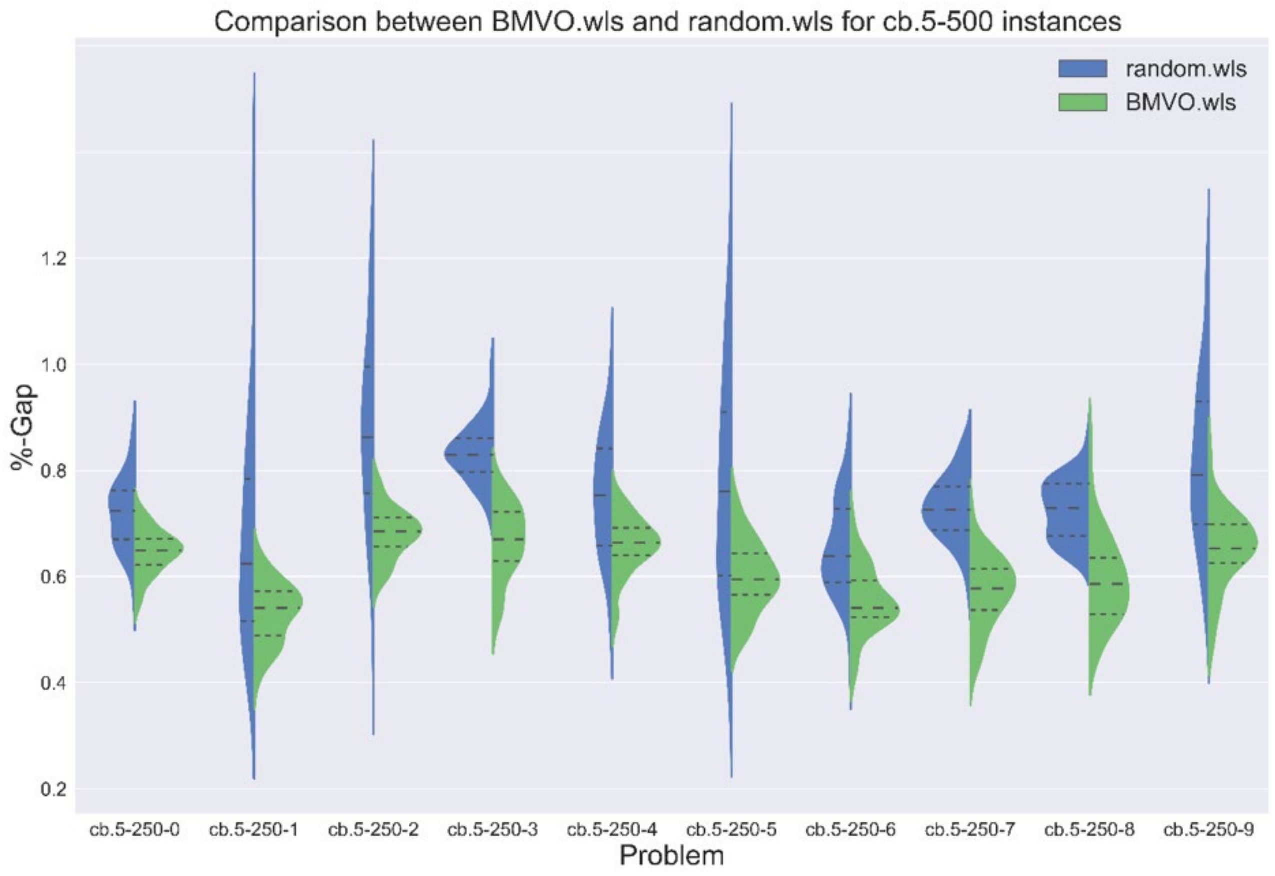

Figure 4. Evaluation of percentile binary operator without Local Search operator

Our next step is trying to separate the contribution of local search operator from the percentile binary operator. For this, we compared the algorithms *wls* and *random.wls*.

When we check the Best Values shown in the **Table 2**, we note that wpe performs better than *05.wpe* in all problems except 1, 6 and 7. However the results are quite close. In the case of the average indicator, wpe outperforms in all problems to *05.wpe*. The Wilcoxon test indicates that the difference is significant. This suggests that wpe is consistently better than *05.wpe*. In the violin chart shown in the **Figure 4** it is further observed that the dispersion of the solutions for the case of *05.wpe* is much larger than in the case of *wpe*. This indicates that the percentile binary operator plays an important role in the precision of the results.

BMVO Comparisons

In this section we evaluate the performance of our BMVO with the algorithm BAAA developed in Zhang et al. (2016). BAAA uses transfer functions as a general mechanism of binarization. In particular BAAA used the $\tanh = \frac{e^{t|x|}-1}{e^{t|x|}+1}$ function to perform the transference. The parameter τ of the tanh function was set to a value 1.5. Additionally, a elite local search procedure was used by BAAA to improve solutions. As maximum number of iterations BAAA used 35000. The computer configuration used to run the BAAA algorithm was: PC Intel Core(TM) 2 dual CPU Q9300@2.5GHz, 4GB RAM and 64-bit Windows 7 operating system. In our BMVO algorithm, the configurations are the same used in the previous experiments. These are described in the **Table 1**. Additionally we made the comparison with KMTR-BH and KMTR-Cuckoo binarizations. KMTR uses the unsupervised K-means learning technique to perform the binarization process. In the article García et al. (2018b), the Black Hole and Cuckoo Search algorithms were binarized using KMTR.

The results are shown in **Table 3**. The comparison was per- formed for the set cb.5.500 of the OR-library. The results for BMVO were obtained from 30 executions for each problem. In black, the best results are marked for both indicators the Best Value and the Average. For the best value indicator BAAA was higher in 4, KMTR-BH in 11, KMTR-Cuckoo in 7 and BMVO in 11. We should note that the sum is greater than 30 because in some cases there was a tie between some of the algorithms. In the Average indicator BAAA was higher in 2 instances, KMTR-BH in 9, KMTR-Cuckoo in 4 and BMVO in 15. We should also note that the standard deviation in most problems was quite low, indicating that BMVO has good accuracy.

Table 3. OR-Library benchmarks MKP CB.5.500

Instance	Best Known	BAAA Best	Avg	KMTR-BH Best	Avg	KMTR-Cuckoo Best	Avg	BMVO Best	Avg	Time(s)	std
0	120148	120066	120013.7	**120096**	120029.9	120082	**120036.8**	120082	120022.6	438	37.1
1	117879	117702	117560.5	**117730**	**117617.5**	117656	117570.6	117656	117617.2	486	43.6
2	121131	120951	120782.9	**121039**	**120937.9**	120923	120855.1	120923	120891.1	457	47.9
3	120804	120572	120340.6	120683	**120522.8**	120683	120455.7	**120683**	120516.4	514	49.1
4	122319	122231	122101.8	**122280**	**122165.2**	122212	122136.4	122212	122134.2	521	49.6
5	122024	121957	121741.8	121982	**121868.7**	121946	121824.6	**121982**	121856.3	531	52.1
6	119127	**119070**	118913.4	119068	**118950.0**	118956	118895.5	118956	118923.1	487	27.8
7	120568	120472	120331.2	120463	**120336.6**	120392	120320.4	**120487**	120317.4	443	66.9
8	121586	121052	120683.6	**121377**	121161.9	121201	121126.3	121295	**121201.4**	431	55.8
9	120717	120499	120296.3	**120524**	120362.9	120467	120335.5	120467	**120391.1**	465	42.1
10	218428	218185	217984.7	218296	218163.7	218291	**218208.9**	**218291**	218203.1	437	30.9
11	221202	220852	220527.5	220951	220813.9	220969	**220862.3**	**220951**	220842.1	436	50.1
12	217542	217258	217056.7	217349	217254.3	217356	217293.0	**217356**	**217297.1**	441	46.1
13	223560	223510	223450.9	**223518**	223455.2	223516	223455.6	223516	**223458.1**	437	42.3
14	218966	218811	218634.3	218848	218771.5	**218884**	218794.0	218848	**218814.4**	471	32.7
15	220530	220429	**220375.9**	**220441**	220342.2	220433	220352.7	220410	220362.7	442	36.1
16	219989	219785	219619.3	219858	219717.9	**219943**	219732.8	219858	**219767.2**	431	57.2
17	218215	**218032**	217813.2	218010	217890.1	218094	217928.7	218010	**217957.1**	428	46.2
18	216976	**216940**	**216862.0**	216866	216798.8	216873	216829.8	216866	216823.1	418	28.2
19	219719	219602	219435.1	219631	219520.0	**219693**	219558.9	219631	**219581.3**	399	31.6
20	295828	295652	295505.0	**295717**	295628.4	295688	295608.8	295688	**295643.1**	389	22.7
21	308086	307783	307577.5	307924	307860.6	**308065**	**307914.8**	307924	307889.1	365	22.4
22	299796	299727	299664.1	299796	**299717.8**	299684	299660.9	**299796**	299713.2	321	33.2
23	306480	306469	306385.0	**306480**	**306445.2**	306415	306397.3	306415	306382.1	327	23.1
24	300342	300240	300136.7	**300245**	300202.5	300207	300184.4	**300245**	**300223.2**	273	27.5
25	302571	**302492**	302376.0	302481	302442.3	302474	302435.6	302481	**302480.2**	347	24.8
26	301339	301272	301158.0	301284	301238.3	**301284**	301239.7	**301284**	**301249.1**	357	20.5
27	306454	306290	306138.4	306325	306264.2	**306331**	306276.4	**306331**	**306308.1**	327	21.1
28	302828	302769	302690.1	302749	302721.4	**302781**	302716.9	302771	**302731.2**	337	25.3
29	299910	299757	299702.3	299774	299722.7	**299828**	299766.0	**299828**	**299771.2**	316	47.1
Average	214168.8	214014.2	213862.0	214059.5	213964.1	214044.2	213959.1	214041.4	213978.9	415.7	38.0

Table 4. Summary of OR-Library benchmarks MKP CB.10.500 and CB.30.500

Problem Set	KMTR-BH Average %-Gap	std	KMTR-Cuckoo Average %-Gap	std	BMVO Average %-Gap	std
cb.10.500.25	0.34	0.08	0.37	0.1	0.39	0.12
cb.10.500.50	0.22	0.03	0.20	0.03	0.21	0.06
cb.10.500.75	0.11	0.03	0.09	0.03	0.10	0.04
cb.30.500.25	0.62	0.12	0.59	0.10	0.64	0.09
cb.30.500.50	0.27	0.05	0.26	0.05	0.24	0.04
cb.30.500.75	0.17	0.03	0.16	0.03	0.18	0.03
Average	0.24	0.1	245.6	0.22	0.29	0.06

Additionally, to evaluate the robustness of BMVO, we experiment with the problems cb.10.500 and cb.30.500. These problems correspond to the most difficult problems of the OR- library. A summary of the results are shown in Table 4. In this table we also incorporate the results for KMTR-BH and KMTR-Cuckoo algorithm.

CONCLUSIONS

In this article, a general binarization technique which use the percentile concept is used to perform the binarization of the MVO algorithm. It should be noted that the percentile technique can be applied in the binarization of any continuous swarm-intelligence algorithm. To test the binarization obtained, we used the knapsack problem to evaluate our binary algorithm. The contribution of the percentile binary operator was studied, observing that this operator contributes to the precision and quality of the solutions obtained. Improving the interquartile range of solutions and best values when we compare the percentile algorithm against a random operator. Additionally, we develop a comparison with the BAAA and KMTR algorithms, showing that BMVO has a good performance.

As a future line of research, it is interesting to compare the performance of the percentile operator with the K-means operator used in KMTR and with the transfer functions used in BAAA, all applied in the binarization of the MVO algorithm. Another interesting line of research is to explore adaptive techniques to automate the selection of parameters. From the theoretical point of view, it would also be interesting to understand how the exploration and exploitation of space are altered when we present the percentile operator. Also, we believe that the incorporation of reinforcement learning techniques for decision making in the binarization mechanism can be an interesting line to explore and that integrates two areas that are developing strongly. Finally, it is interesting to use the percentile operator to binarize other swarm intelligence algorithms in addition to solving other NP-hard problems.

REFERENCES

Astorga, G., Crawford, B., Soto, R., Monfroy, E., García, J. and Cortes, E. (2018). A meta-optimization approach to solve the set covering problem. *Ingeniería, 23*(3), 1-14.

Bansal, J. C. and Deep, K. (2012). A modified binary particle swarm optimization for knapsack problems. *Applied Mathematics and Computation, 218*(22), 11042-11061. https://doi.org/10.1016/j.amc.2012.05.001

Barman, S. and Kwon, Y. K. (2017). A novel mutual information-based Boolean network inference method from time-series gene expression data. *PloS one, 12*(2), e0171097. https://doi.org/10.1371/journal.pone.0171097

Crawford, B., Soto, R., Astorga, G., García, J., Castro, C. and Paredes, F. (2017). Putting continuous metaheuristics to work in binary search spaces. *Complexity*. https://doi.org/10.1155/2017/8404231

Crawford, B., Soto, R., Monfroy, E., Astorga, G., García, J. and Cortes, E. (2018). A meta-optimization approach to solve the set covering problem. *Ingeniería, 23*(3), 274-288. https://doi.org/10.14483/23448393.13247

Crawford, B., Soto, R., Monfroy, E., Astorga, G., García, J. and Cortes, E. (2017, September). A meta-optimization approach for covering problems in facility location. In *Workshop on Engineering Applications* (pp. 565-578). Springer, Cham. https://doi.org/10.1007/978-3-319-66963-2_50

Garcia, J. and Măntoiu, M. (2014). Localization results for zero order pseudodifferential operators. *Journal of Pseudo-Differential Operators and Applications, 5*(2), 255-276. https://doi.org/10.1007/s11868-013-0084-y

García, J. and Peña, A. (2018). Robust optimization: concepts and applications. Nature-inspired Methods for Stochastic, *Robust and Dynamic Optimization, 7*. https://doi.org/10.5772/intechopen.75381

García, J., Altimiras, F., Peña, A., Astorga, G. and Peredo, O. (2018a). A binary cuckoo search big data algorithm applied to large-scale crew scheduling problems. *Complexity*. https://doi.org/10.1155/2018/8395193

García, J., Crawford, B., Soto, R. and Astorga, G. (2017, September). A percentile transition ranking algorithm applied to knapsack problem. In *Proceedings of the Computational Methods in Systems and Software* (pp. 126-138). Springer, Cham. https://doi.org/10.1007/978-3-319-67621-0_11

García, J., Crawford, B., Soto, R. and Astorga, G. (2019a). A clustering algorithm applied to the binarization of swarm intelligence continuous metaheuristics. *Swarm and evolutionary computation, 44*, 646-664. https://doi.org/10.1016/j.swevo.2018.08.006

García, J., Crawford, B., Soto, R. and García, P. (2017, February). A multi dynamic binary black hole algorithm applied to set covering problem. In *International Conference on Harmony Search Algorithm* (pp. 42-51). Springer, Singapore. https://doi.org/10.1007/978-981-10-3728-3_6

García, J., Crawford, B., Soto, R., Castro, C. and Paredes, F. (2018b). A k-means binarization framework applied to multidimensional knapsack problem. *Applied Intelligence, 48*(2), 357-380. https://doi.org/10.1007/s10489-017-0972-6

García, J., Moraga, P., Valenzuela, M., Crawford, B., Soto, R., Pinto, H., ... Astorga, G. (2019b). A Db-Scan Binarization Algorithm Applied to Matrix Covering Problems. *Computational intelligence and neuroscience*. https://doi.org/10.1155/2019/3238574

García, J., Pope, C. and Altimiras, F. (2017). A Distributed-Means Segmentation Algorithm Applied to Lobesia botrana Recognition. *Complexity*. https://doi.org/10.1155/2017/5137317

Graells-Garrido, E. and García, J. (2015, December). Visual exploration of urban dynamics using mobile data. In *International Conference on Ubiquitous Computing and Ambient Intelligence* (pp. 480-491). Springer, Cham. https://doi.org/10.1007/978-3-319-26401-1_45

Graells-Garrido, E., Peredo, O. and García, J. (2016). Sensing urban patterns with antenna mappings: the case of Santiago, Chile. *Sensors, 16*(7), 1098. https://doi.org/10.3390/s16071098

Haddar, B., Khemakhem, M., Hanafi, S. and Wilbaut, C. (2016). A hybrid quantum particle swarm optimization for the multidimensional knapsack problem. *Engineering Applications of Artificial Intelligence, 55*, 1-13. https://doi.org/10.1016/j.engappai.2016.05.006

Kong, X., Gao, L., Ouyang, H. and Li, S. (2015). Solving large-scale multidimensional knapsack problems with a new binary harmony search algorithm. *Computers & Operations Research, 63*, 7-22. https://doi.org/10.1016/j.cor.2015.04.018

Kumar, A. and Suhag, S. (2017). Multiverse optimized fuzzy-PID controller with a derivative filter for load frequency control of multisource hydrothermal power system. *Turkish Journal of Electrical Engineering & Computer Sciences, 25*(5), 4187-4199. https://doi.org/10.3906/elk-1612-176

Liu, J., Wu, C., Cao, J., Wang, X. and Teo, K. L. (2016). A binary differential search algorithm for the 0–1 multidimensional knapsack problem. *Applied Mathematical Modelling, 40*(23-24), 9788-9805. https://doi.org/10.1016/j.apm.2016.06.002

Meng, T. and Pan, Q. K. (2017). An improved fruit fly optimization algorithm for solving the multidimensional knapsack problem. *Applied Soft Computing, 50*, 79-93. https://doi.org/10.1016/j.asoc.2016.11.023

Mirjalili, S., Mirjalili, S. M. and Hatamlou, A. (2016). Multi-verse optimizer: a nature-inspired algorithm for global optimization. *Neural Computing and Applications, 27*(2), 495-513. https://doi.org/10.1007/s00521-015-1870-7

Peredo, O. F., García, J. A., Stuven, R. and Ortiz, J. M. (2017). Urban dynamic estimation using mobile phone logs and locally varying anisotropy. In *Geostatistics Valencia 2016* (pp. 949-964). Springer, Cham. https://doi.org/10.1007/978-3-319-46819-8_66

Pirkul, H. (1987). A heuristic solution procedure for the multiconstraint zero ‐ one knapsack problem. *Naval Research Logistics (NRL), 34*(2), 161-172. https://doi.org/10.1002/1520-6750(198704)34:2<161::AID-NAV3220340203>3.0.CO;2-A

Trivedi, I. N., Jangir, P., Jangir, N., Parmar, S. A., Bhoye, M. and Kumar, A. (2016, March). Voltage stability enhancement and voltage deviation minimization using multi-verse optimizer algorithm. In *2016 International conference on circuit, power and computing technologies (ICCPCT)* (pp. 1-5). IEEE. https://doi.org/10.1109/ICCPCT.2016.7530136

Valenzuela, M., Valenzuela, P., Cáceres, C., Jorquera, L. and Pinto, H. (2019, June). A Percentile Multi-Verse Optimizer Algorithm applied to the Knapsack problem. In *2019 14th Iberian Conference on Information Systems and Technologies (CISTI)* (pp. 1-8). IEEE. https://doi.org/10.23919/CISTI.2019.8760613

Zhang, X., Wu, C., Li, J., Wang, X., Yang, Z., Lee, J. M. and Jung, K. H. (2016). Binary artificial algae algorithm for multidimensional knapsack problems. *Applied Soft Computing, 43*, 583-595. https://doi.org/10.1016/j.asoc.2016.02.027