RAIRO-Oper. Res. 60 (2026) 1931–1957
https://doi.org/10.1051/ro/2026050

RAIRO Operations Research
www.rairo-ro.org

BRKGA APPLIED TO THE CLUSTER ENSEMBLE PROBLEM

Augusto Pizano Vieira Beltrão[1,*], Luiz Satoru Ochi[1], José André de Moura Brito[2], Gustavo Silva Semaan[3] and Augusto César Fadel[4]

Abstract. Clustering algorithms are used to partition datasets associated with various real-world applications. However, in addition to the adopted algorithm, the obtained partition depends on the data distribution. Consequently, applying a single algorithm can lead to producing a poor-quality partition. Cluster Ensemble (CE) is an alternative to produce a good quality partition, as it combines different dataset partitions into a single consensus partition. According to the literature, in general, the consensus partition is less sensitive to noise when compared to that produced by a single algorithm. This work proposes a CE algorithm (BRKGA-CE) that combines: (i) three different strategies for producing base partitions; (ii) the BRKGA metaheuristic; (iii) the mean silhouette index, and (iv) an iterative method applied in the final phase of BRKGA-CE that allocates each object into a cluster. The core idea of BRKGA-CE is to find the representative objects of each cluster in order to maximize the mean silhouette index. To evaluate BRKGA-CE, computational experiments were carried out with 20 datasets and the main algorithms from the literature, where two well-known external validation indices ($\mathcal{NMI}$ and $\mathcal{AR}$) and hypothesis tests were applied. As a result, BRKGA-CE presented good-quality solutions, in the three strategies compared to the other algorithms.

Mathematics Subject Classification. 90C59, 62H30.

Received December 30, 2023. Accepted May 8, 2026.

1. Introduction

There are many real applications whose solution is associated with solving a cluster analysis (CA) problem [8, 37]. In general, when solving a CA problem, the objective is to divide the n objects of a dataset X into k clusters defined by $\{C_1, \ldots, C_r, \ldots, C_k\}$, corresponding to a partition so that objects from the same cluster are similar to each other (according to some similarity metric), while objects from different clusters are dissimilar to each other [8]. CA applications are found, for example, in Social Sciences, Marketing, Statistics, and Medicine [37].

Keywords. Cluster ensemble, consensus clustering, optimization, cluster analysis.

[1] Instituto de Computação da Universidade Federal Fluminense, Niterói 24210-310, Brasil.
[2] Escola Nacional de Ciências Estatísticas, Rio de Janeiro 20231-050, Brasil.
[3] Universidade Federal Rural do Rio de Janeiro, Três Rios 25802-100, Brasil.
[4] Instituto Brasileiro de Geografia e Estatística, Rio de Janeiro 20021-060, Brasil.
*Corresponding author: apizano@id.uff.br

According to [8], the classic clustering algorithms (non-hierarchical and hierarchical) have limitations, generally because they produce partitions that depend on the distribution of objects (cluster structure) in the dataset.

An alternative considered in the literature to deal with this issue and, at the same time, produce a good quality and robust partition – in the light of some validation index (regardless of the dataset cluster structure), concerns the use of the Cluster Ensemble (CE) [49]. In the CE, a set composed of multiple base partitions (ensemble) is combined (through the application of a consensus function) into a unique solution – called consensus partition. In order to produce a good-quality partition, there needs to be diversity in the base partitions; this characteristic can be obtained by producing such partitions through the application of different clustering algorithms or using the same algorithm but by varying the value of one or more of its parameters, such as the number of clusters [8]. An advantage of using CE is that, in general, the consensus partition has the following desirable properties [18,50,51,53]: (i) robustness [53]: better average performance[1] than clustering algorithms separately; (ii) consistency [53]: the partition resulting from the combination must be similar to the base partitions of the ensemble. The similarity is generally calculated from some external validation index; (iii) novelty [53]: CE tends to produce a combined solution unattainable by a single clustering algorithm; (iv) stability [50,53]: less sensitivity to noise and *outliers*.

In this work, the resolution of the Clustering Ensemble Problem (CEP) – obtaining the consensus partition – is made by an algorithm based on the Biased Random Key Genetic Algorithm (BRKGA) metaheuristic [46], also applied with success in other clustering problems [10, 13, 38] from the literature. A review paper presents several applications of BRKGA in optimization problems, and its advantages, such as fast convergence to high-quality solutions [39]. The approach proposed in this work has the following differentials: (i) use of the BRKGA metaheuristic to solve the CE; (ii) application of three different strategies to produce the base partitions, allowing the diversity of solutions; (iii) use of the average silhouette index to select the best solutions, calculated from a co-association matrix (replacing the traditional Euclidean distance matrix); (iv) production of good quality partitions compared with state-of-the-art clustering ensemble algorithms, with the results being validated through the application of two external validation indexes (Adjusted Rand Index and Normalized Mutual Information) and hypothesis tests.

The algorithm is applied in three phases. Basically, in the first phase, an ensemble of base partitions is produced from several classic clustering algorithms, considering the adoption of three strategies that include the application of non-hierarchical, hierarchical, or both types of algorithms. In the second phase, the algorithm selects the best set of k objects from the dataset to represent the k clusters (the set of objects that produces the partition with the highest mean silhouette index value calculated from the co-association matrix). In the third phase, the remaining objects are allocated iteratively to one of the clusters until all objects are in some cluster, thus producing the consensus partition.

It is expected that the proposed algorithm is able to obtain high quality solutions (according to validation criteria commonly used in the literature) in relation to other known methods.

The main contributions of this work are listed below:

- We propose a new algorithm called Biased Random Key Genetic Algorithm Cluster Ensemble (BRKGA-CE), based on the the BRKGA metaheuristic. The objective of this algorithm is to select representative objects in order to maximize the silhouette index and, consequently, produce better quality partitions, in light of the Adjusted Rand Index ($\mathcal{AR}$) and Normalized Mutual Information ($\mathcal{NMI}$) indexes.
- In computational experiments, three base partition generation strategies are used to evaluate the performance of algorithms from the literature as well as BRKGA-CE in different scenarios. BRKGA-CE presented good results in the 3 scenarios evaluated.

In addition to the introduction, this paper is divided into four sections. The second section brings the description of the CEP and the literature review. In the third section, the proposed algorithm is described, and

[1]The consensus partition must have high goodness of fit with the true partition (*ground truth*). In this case, the true partition corresponds to the classes (labels) of objects in classification datasets, when available [18].

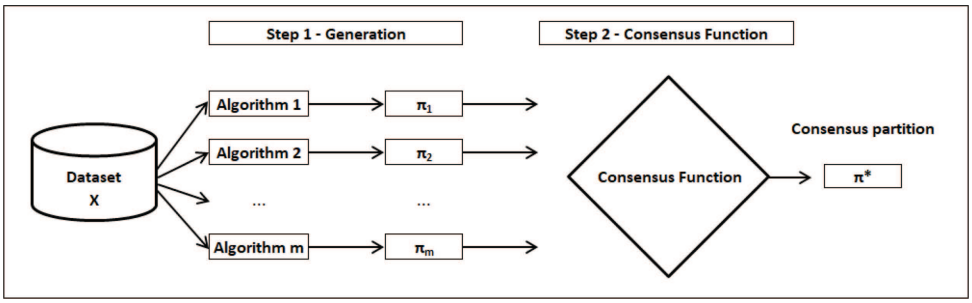

FIGURE 1. Main steps of the cluster ensemble.

in the fourth section, the results of the experiments carried out comparing the algorithm with others in the literature are presented. Finally, the fifth section contains conclusions and future work.

2. CLUSTER ENSEMBLE PROBLEM

Let $X = \{x_1, \ldots, x_i, \ldots, x_n\}$ be a dataset consisting of n objects, where each object $x_i \in X$ is represented by a vector with d attributes of the quantitative, qualitative or mixed type, such that $x_i = (x_{i1}, \ldots, x_{id})$. Additionally, consider a cluster ensemble composed of m base partitions (base clusterings) defined by $\Pi = \{\pi_1, \ldots, \pi_g, \ldots, \pi_m\}$, with each partition π_g associated with a set defined by k_g clusters, that is:
$\pi_g = \{C_1^g, \ldots, C_{k_g}^g\}$, such that $\cup_{r=1}^{k_g} C_r^g = X$ and $C_r^g \cap C_u^g = \emptyset, \forall\, C_r^g, C_u^g \in \pi_g$.

For each $x_i \in X$, $\psi^g(x_i)$ denotes the cluster label of x_i in the base partition π_g, that is, the function ψ^g serves to indicate which cluster each object is allocated in each base partition. When solving the CEP, the objective is to find a new partition (consensus partition) $\pi^* = \{C_1^*, \ldots, C_k^*\}$ of X of good quality (based on a validation index), which summarizes the information from the ensemble Π. Basically, the new partition π^* is obtained from the application of the consensus function in the ensemble Π, that combines the base partitions into a single consensus partition. In this work, the generation of the base partitions used is based on the adoption of three strategies that combine several clustering algorithms, while the consensus function was incorporated into a BRKGA described in Section 3. Figure 1 presents an overview of the CEP.

As already mentioned, the principle of the Cluster Ensemble is to combine multiple base partitions into a consensus partition through a consensus function [26, 49]. The survey article by [8] classifies most approaches that implement CE consensus functions into four broad categories, namely:

(i) Direct approach: The allocation of objects to clusters is made by voting criteria [3, 4, 51]. As this is an unsupervised learning problem, there is a correspondence problem between labels,[2] since the specific label used for a given cluster on each base partition, is arbitrary. Generally, methods based on the direct approach have, as a restriction, that all base partitions have the same number of clusters. In [9], the search for correspondence between the cluster labels of different partitions is treated as an optimization problem.

(ii) Feature-based approach: This category uses an array of labels (which correspond to nominal data) [45, 52], disregarding the search for correspondence between the labels of different base partitions. Some of these methods approach the CEP as an optimization problem using a consensus function based on the Median Partition Problem (MPP). In this case, we seek to maximize the similarity (or minimize the dissimilarity) between the consensus partition and the base partitions. When Mirkin's metric [43] is used, MPP is $\mathcal{NP}$-hard [53]. In [15], a local search named Best-One-element-Move (BOM) is proposed to solve MPP.

(iii) Pairwise-similarity based approach: This category uses the co-association matrix, denoted by CO $(n \times n)$, which summarizes the information of the base partitions, more specifically, indicates the similarity

[2]Objects with the same label belong to the same cluster, and objects with different labels belong to different clusters.

between X object pairs. Each entry $\mathrm{CO}(x_i, x_j)$ corresponds to the ratio between the number of times (n_{ij}) that objects x_i and x_j belong to the same cluster among the m partitions and the total number of base partitions [18]:

$$\mathrm{CO}(x_i, x_j) = \frac{n_{ij}}{m}, \ \forall\, i, j = 1, \ldots, n \ (i \neq j), \tag{1}$$

where $\mathrm{CO}(x_i, x_j) = \mathrm{CO}(x_j, x_i)$ and $\mathrm{CO}(x_i, x_i) = 1$.

It is possible to apply clustering algorithms considering CO as a similarity matrix[3] to produce the consensus partition. In this sense, Fred and Jain [18] used applied hierarchical algorithms in the CO matrix to produce the consensus partition. This approach is the evidence accumulation clustering (EAC). Iam-On and Boongoen [31] presented three alternatives to use in place of the CO matrix (CTS, SRS, ASRS), which can be used with hierarchical or graph partitioning algorithms. Huang *et al.* [25] considered different weights (weighted) for each base partition in the calculation of the CO matrix, the proposed consensus function is termed weighted evidence accumulation clustering (WEAC). In [26], a novel consensus functions, termed locally weighted evidence accumulation (LWEA), uses different weights were considered for each cluster of each base partition; in [27], in addition to CO, the information was considered at higher granularity levels (intersection of clusters). Some papers use the fuzzy logic [2, 14].

(iv) Graph-based approach: This category represents the ensemble information through graphs. Initially, a weighted graph is constructed from the base partitions. Generally, the vertices represent the objects of X, and the edges are the connection (similarity) between them (calculated from the base partitions). One possibility for the calculation of similarities is to consider the same value of the co-association matrix. This graph is then partitioned into k clusters to produce the consensus partition [32, 49, 58], applying some graph partitioning technique.

Other works: The authors of [28] proposed an approach for solving the CEP on large datasets (up to 20 million objects) through *spectral clustering* and the work of [7] combines multiple partitions generated by *k-means* to identify non-linearly separable clusters.

The authors of [29] paper proposed a novel ensemble clustering framework named MDEC that uses wide range of diversified metrics, random subspaces, and weighted clusters in order to solve cluster ensemble for datasets with high-dimensional data. Experiments are conducted on 30 high-dimensional datasets comparing proposed algorithms (MDEC-HC, MDEC-SC, and MDEC-BG) with 9 algorithms from literature.

The authors of [61] proposed the TRCE (Tri-level Robust Clustering Ensemble) algorithm that transforms the CEP into a multiple graph learning problem. TRCE seeks to build a consensus partition considering robustness at 3 levels: Base Clustering, Graph and Instance Level of Robustness. The experiments were carried out considering only 10 datasets, comparing TRCE to 10 literature methods. Each ensemble is composed of 20 base partitions generated using k-means.

The authors of [62] propose a new method via structured hypergraph learning named CESHL to solve CEP. This method is based on optimizing an objective function. The authors impose some constraints in order to make the hypergraph will have exact k connected components so obtaining the consensus partition is trivial.

The authors of [35] proposed a new cluster ensemble model based on a co-association matrix self-enhancement framework. This model uses a high-confidence information sparse matrix HC obtained from the base partitions (which is a highly-reliable pairwise information from the objects). The proposed method uses simultaneously the information in the HC matrix and the original co-association matrix in order to learn an improved co-association matrix to achieve better performance. Experiments are carried out in 8 datasets comparing the proposed method with 12 cluster ensemble methods from the literature.

The authors of [63] proposed a novel Self-paced Consensus Clustering with Adaptive Bipartite Graph (SCCABG) to gradually involve data from the more reliable ones to the less reliable in order to obtain the

[3]The complement of the matrix CO can be used as a distance (dissimilarity) matrix (in this work, called $D_{\mathrm{CO}} = 1 - \mathrm{CO}$). That is, instead of considering that the dissimilarity between two objects x_i and x_j corresponds to, for example, the Euclidean distance between x_i and x_j, it is considered that this distance corresponds to the proportion of times x_i and x_j are not allocated to the same cluster in the base partitions.

consensus partition. An bipartite graph is constructed from the base partitions (nodes are clusters and instances, edges indicate that an instance belongs to a cluster). Then, the model learn a structured bipartite graph from this initial one by self-paced learning. The final consensus partition is obtained from the learned structured bipartite graph. The proposed method is compared with 12 algorithms from the literature in 8 datasets. The ensemble is composed of 20 base partitions generated by k-means.

The authors of [55] combined rough set theory with cluster ensemble to propose the Dual-granularity weighted ensemble clustering (DGWEC). The consensus partition is constructed by using an hierarchical clustering algorithm in a new weighted co-association matrix (which is based on global cluster reliability and local sample pair similarity). The experiments are conducted in 30 datasets with an ensemble generated from 50 runs of k-means.

The paper [19] is a review of the recent developments of ensemble deep learning models and cites consensus clustering (unsupervised learning).

Cluster ensemble and genetic algorithms: Several works in the literature combine Cluster Ensemble and genetic algorithms (GAs). In this sense, a review on accuracy optimization[4] involving Cluster Ensemble and GAs can be found in [21].

The authors of [30] propose a GA that seeks to minimize the sum of dissimilarities (based on the entropy of Information Theory) between the consensus partition and the base partitions. In [6], a GA is proposed to maximize an objective function that considers the co-association between objects within the same cluster and between objects from different clusters.

The authors of [24] propose a GA that uses a consensus function as a crossover operator, that produces a co-association matrix from the partitions and applies the hierarchical algorithm of average linkage on this matrix to obtain a new partition. The objective function to be minimized corresponds to the distance between each object and the centroid of its cluster.

The authors of [12] have developed a multiobjective genetic algorithm combined with the Cluster Ensemble that uses two indexes as an optimization criterion and in [57] is proposed a GA called HGMCE, which uses a new objective function where each base partition is considered a new data attribute.

Computational experiments: The algorithms used for comparison in the experiments are: BOM [15], EAC (EAC-SL, EAC-AL, EAC-CL) [18], SRS (SRS-SL, SRS-AL, SRS-CL) [31, 32], CTS (CTS-SL, CTS-AL, CTS-CL) [31, 32], ASRS (ASRS-SL, ASRS-AL, ASRS-CL) [31], WEAC (WEAC-SL, WEAC-AL, WEAC-CL) [25] and LWEA (LWEA-SL, LWEA-AL, LWEA- CL) [26].

Table 1 provides a comparative summary of the four main categories of consensus functions and positions the proposed BRKGA-CE within this context. Each category presents different trade-offs between flexibility, computational cost, and the quality of the resulting consensus partitions. Direct and feature-based approaches are simple and efficient but often rely on assumptions such as a fixed number of clusters or label correspondence among base partitions, which reduces their adaptability. Pairwise-similarity and graph-based techniques are effective at capturing complex relationships among data points, although they can be more computationally demanding and may face challenges when applied to very large datasets. Distinct from these methods, BRKGA-CE employs a metaheuristic-based optimization framework capable of exploring new regions of the solution space and directly maximizing a quality measure, such as the mean silhouette. Despite its higher computational effort, this approach offers greater flexibility to integrate heterogeneous base partitions and to produce robust, high-quality consensus solutions

[4]Accuracy is a term often associated with supervised learning but it can also be used to calculate the quality of unsupervised classification solutions (*i.e.* partition generated by a clustering algorithm) [21, 49]. The calculation of accuracy (and other measures associated with the classification task) uses the class label considering classification datasets.

TABLE 1. Main categories of CE approaches.

Categories	Consensus function	Characteristics
Direct approach	Simple Voting [51] Incremental Voting [4] LCS [9]	The allocation of objects to clusters is done based on voting criteria. As this is an unsupervised learning problem, there is a correspondence issue between labels. There is generally an assumption that the number of clusters is the same for the consensus partition and the base partitions.
Feature-based approach	Mixture Model [52] Iterative Voting Consensus (IVC) [45] BOM [15]	This category uses arrays of labels (nominal data), disregarding the search for correspondence between labels of different base partitions. The number of clusters for the consensus partition and for the base partitions is not necessarily the same.
Pairwise-similarity based approach	Evidence Accumulation Clustering (EAC) [18] CTS, SRS, ASRS [31] Weighted EAC [25] LWEA [26] ECPCS [27]	Represents ensemble information through a variant of the co-association matrix (CO). Clustering algorithms can be applied using CO as a similarity matrix to produce the consensus partition. The number of clusters for the consensus partition and for the base partitions is not necessarily the same.
Graph-based approach	SRS [32] CSPA [49] HGPA [49] MCLA [49] GCC [58]	Represents ensemble information through a weighted graph, which is then partitioned into k clusters using a graph partitioning technique. The number of clusters for the consensus partition and for the base partitions is not necessarily the same.
Proposed consensus function	BRKGA-CE	BRKGA-CE is an iterative metaheuristic approach that can incorporate a local search and represents each solution as a vector of random keys. Its objective function combines the silhouette index and the co-association matrix to define the medoids of the consensus partition. An iterative procedure is applied to construct the consensus partition. One advantage of BRKGA-CE is its ability to explore new search spaces; however, it may be computationally intensive.The number of clusters for the consensus partition and for the base partitions is not necessarily the same.

3. PROPOSED ALGORITHM

3.1. Biased random-key genetic algorithm

In the Biased Random-Key Genetic Algorithm (BRKGA) metaheuristic,[5] each solution to the optimization problem is represented by a Y vector (chromosome), composed of N random keys – values generated according

[5]The Biased Random-Key Genetic Algorithm [22, 23] has been applied to solve several optimization problems. N is the chromosome size; p is the population size; p_e is the elite set size (number of chromosomes in the elite set); p_m is the number of mutant chromosomes.

Algorithm 1. BRKGA.

$f_{\text{best}} \leftarrow +\infty$ (minimization problem)
Generate the initial population $\mathcal{P} = \{Y_1, \ldots, Y_p\}$
$q \leftarrow 0$
While $(q < \max_{\text{gen}})$ **do**
  $q \leftarrow q + 1$
  Apply the decoder to each $Y_t \in \mathcal{P},\ t = 1, \ldots, p$
  Calculate the value of $fobj$ for each decoded solution
  Sort chromosomes according to $fobj$ value
  Partition the chromosomes of P into Y_e and Y_{ne}
  Select the best chromosome $Y_{\text{new}} \in Y_e$
  if $fobj(Y_{\text{new}}) < f_{\text{best}}$ **then**
    $Y_{\text{best}} \leftarrow Y_{\text{new}}$; $f_{\text{best}} \leftarrow fobj(Y_{\text{new}})$;
  Copy chromosomes from Y_e to $\mathcal{P}'$
  Generate p_m mutant chromosomes and add them to $\mathcal{P}'$
  Apply crossover using Y_e and Y_{ne}
  Add the offspring chromosomes in $\mathcal{P}'$
  $\mathcal{P} \leftarrow \mathcal{P}'$
End-While

to $U[0, 1]$. Each Y vector is transformed into a solution to the optimization problem addressed, applying a decoder (problem-specific procedure), which returns a feasible solution and its associated objective function value ($fobj$). In the first generation of BRKGA, p vectors of random keys are produced, which make up the initial population, denoted by $\mathcal{P} = \{Y_1, \ldots, Y_p\}$. In subsequent generations, the population is updated by applying the genetic operators: selection, mutation, and crossover. Traditional Genetic Algorithms (GAs) represent candidate solutions directly, typically using binary or real-valued encodings, and rely on standard selection, crossover, and mutation operators [42]. By contrast, BRKGA employs a biased random-key representation, in which each solution is encoded as a vector of random keys that are decoded into feasible solutions [39]. This encoder/decoder process allows BRKGA to efficiently explore large search spaces and incorporate problem-specific heuristics. Consequently, BRKGA often converges faster and produces higher-quality solutions than standard GAs, especially for complex optimization problems.

Selection: The p chromosomes of $\mathcal{P}$ are partitioned into two sets: elite set Y_e (the size of the elite set is equal to p_e, where $p_e < p/2$) [46] formed by the best chromosomes, according to the value of the objective function and non-elite set Y_{ne} (size $p - p_e$), formed by the remaining chromosomes. The chromosomes from Y_e are copied to the next generation's population, denoted by $\mathcal{P}'$.

Mutation: Analogously to the p chromosomes generated in the initial population, p_m mutant chromosomes are generated.

Crossover: $(p - p_e - p_m)$ chromosomes are produced. More specifically, in each crossover,[6] a chromosome from Y_e and a chromosome from Y_{ne} are selected to produce a new chromosome (offspring). Each offspring chromosome's random key has probability $\rho > 0.5$ of inheriting the random key from Y_e and probability $1 - \rho$ of inheriting the random key from Y_{ne}.

After applying these operators to the population $\mathcal{P}$, the population of the next generation ($\mathcal{P}'$) will be composed by p_e chromosomes from Y_e, p_m mutant chromosomes and $(p - p_e - p_m)$ chromosomes obtained from the crossover. BRKGA is executed until some stopping criterion is satisfied, such as the maximum number of generations or processing time. Algorithm 1 presents the general pseudocode of BRKGA, having the maximum number of generations as a stopping criterion.

[6] Using the parametrized uniform crossover [22].

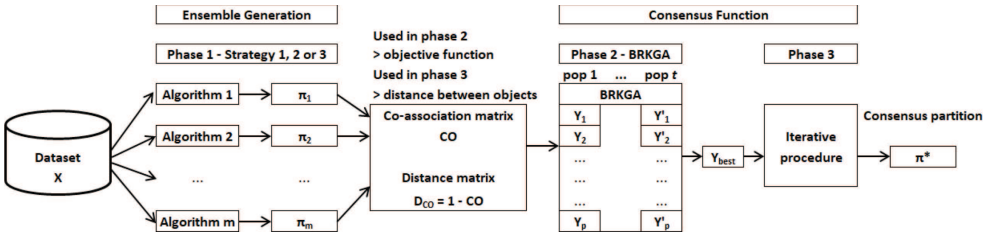

FIGURE 2. Phases of the proposed algorithm.

3.2. Biased random-key genetic algorithm cluster ensemble

The algorithm proposed in this work, Biased Random-Key Genetic Algorithm Cluster Ensemble (BRKGA-CE), is based on the BRKGA metaheuristic. This algorithm is applied in three phases: **(1)** – Construction of m base partitions, considering three possible strategies that include using a set of clustering algorithms. At the end of this phase, the co-association matrix (CO) is calculated, which serves as the basis for calculating the dissimilarity matrix D_{CO} (defined later) that will be used as a distance matrix in phases 2 and 3; **(2)** – Selection of k objects (from X) representative of the k clusters. This selection is made by a BRKGA procedure, where each chromosome represents a possible set of k representative objects.[7] The evaluation of the chromosomes in the BRKGA is done by an objective function associated with the mean silhouette index (calculated using the D_{CO} matrix, see Sect. 3.4.2). At the end of the BRKGA execution, the best chromosome (set of k objects that present the best value of mean silhouette index) of the last generation according to the objective function is selected to be used in phase 3. **(3)** – The k objects selected in phase 2, associated with the best chromosome, are used as a starting point for constructing the consensus partition, with the $(n-k)$ dataset objects allocated iteratively to one of the k clusters (nearest cluster) until all objects are associated with some cluster. The distance between objects and clusters is calculated from the D_{CO} matrix. Figure 2 and the Algorithm 2 summarize how BRKGA-CE works.

Line 1 of the pseudocode (Algorithm 2) corresponds to phase 1. In lines 2 and 3, the matrix D_{CO} is calculated, which will be used as a distance matrix for calculating the objective function in phase 2 and the distance between objects in phase 3. Phase 1 corresponds to the generation of base partitions, and phases 2 and 3 correspond to the consensus function. Line 4 corresponds to phase 2 (application of BRKGA to select a set Y_{best} of k objects from X). Line 5 corresponds to phase 3 (iterative procedure), which uses the set of k representative objects Y_{best} to build the consensus partition π^*. The κ parameter influences the shape of the clusters (elongated or compact) and is explained in the Section 3.5.

Algorithm 2. BRKGA-CE.

1: Generate base partitions: $\Pi \leftarrow \textit{strategy}()$
2: Compute the co-association matrix CO from Π
3: Calculate the matrix of distances $D_{\text{CO}} = 1 - \text{CO}$
4: Select set of representative objects: $Y_{\text{best}} \leftarrow BRKGA(p, p_e, p_m, \rho, k, \Pi, D_{\text{CO}})$
5: Consensus partition: $\pi^* \leftarrow \textit{iterative\_procedure}(Y_{\text{best}}, k, \Pi, \kappa, D_{\text{CO}})$
6: **return** π^*

[7]The representative objects corresponds to the medoids from the k-medoids problem, *i.e.*, serves as the center of its cluster. Each of the remaining $n-k$ objects of X is allocated to the nearest medoid cluster.

3.3. Phase 1: generation of base partitions

In order to guarantee diversity to the base partitions, these partitions are generated by adopting three different strategies, which include the application of some classic clustering algorithms, as shown below:

- **Strategy 1**: non-hierarchical algorithms:
  (i) based on 3 versions of k-means [34, 41]: Hartigan-Wong, Lloyd, MacQueen. Implemented in the base R package.
  (ii) based on 3 versions of k-medoids [37,44]: PAM, CLARA and CLARANS. Implemented in the R package 'fastkmedoids'.
- **Strategy 2**: hierarchal agglomerative algorithms [56]: simple, average, complete, Ward, median, centroid, and *Mcquitty* linkages, implemented in the R base package (function *hclust*). Divisive hierarchical algorithm: DIANA, implemented in the R package 'cluster'.
- **Strategy 3**: considers all hierarchical and non-hierarchical algorithms from Strategies 1 and 2.

In addition to these strategies, in each execution of each algorithm, the value of k is randomly selected from the set $K = \{2, 3, \ldots, \sqrt{n}\}$,[8] as recommended in [17, 26, 32] as a rule of thumb and commonly used in works associated with clustering problems [1, 48].

3.4. Phase 2: BRKGA

3.4.1. Chromosome and decoder

Each chromosome corresponds to a vector of size k, each entry being a real number in the interval [0,1], and each chromosome is mapped into a set of representative objects using the decoder by converting each entry of the chromosome in an identifier corresponding to one of the objects of X (integer number belonging to the set $\{1, \ldots, n\}$), as described next.

Create a set of size n containing all objects from X enumerated by an identifier $id = 1, \ldots, n$. This decoder is applied iteratively and without replacement by selecting an object to represent one of the clusters in each of the k iterations.

Initially (iteration 1), multiply the first chromosome entry by n and apply the *ceiling* function. This integer number obtained corresponds to the identifier of the first representative object, that will not be considered in the next iterations. Thus, the set of objects to be considered in iteration 2 has size $(n-1)$. Next, the second representative object is selected from the second entry of the chromosome, multiplying by $(n-1)$ and applying the *ceiling* function. This number corresponds to the identifier of the second representative object. This process is repeated until k objects are selected from X (which corresponds to the number of chromosome entries).

An example is shown below of mapping a chromosome $Y = \{0.19, 0.87, 0.23\}$ of size $k = 3$ for a dataset with $n = 10$ objects.

- First representative object: Multiply the first entry of Y by n and apply the ceiling function: $\lceil 0.19 \times 10 \rceil = 2$. Select the 2nd object of $\{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}\}$. Thus, the representative object of the first cluster is the object x_2.
- Second representative object: multiply the second entry of Y by $(n-1)$ and apply the ceiling function: $\lceil 0.87 \times 9 \rceil = 8$. Select the 8th object of $\{x_1, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}\}$. Thus, the representative object of the second cluster is the object x_9.
- Third representative object: multiply the third entry of Y by $(n-2)$ and apply the ceiling function: $\lceil 0.23 \times 8 \rceil = 2$. Select the 2nd object of $\{x_1, x_3, x_4, x_5, x_6, x_7, x_8, x_{10}\}$, which implies that the representative object of the third cluster is x_3.

Figure 3 shows the decoding process. The representative objects are x_2, x_9, x_3.

[8] If $\sqrt{n}$ is not an integer, it is rounded down.

Y = {0.19, 0.87, 0.23}
k = 3
n = 10
X = {x_1, ..., x_{10}}

iter 1	**iter 2**	**iter 3**
h = 1	h = 2	h = 3
(n-h+1) = 10	(n-h+1) = 9	(n-h+1) = 8
Y[h] = 0.19	Y[h] = 0.87	Y[h] = 0.23
ceiling(10 * 0.19) = 2	ceiling(9 * 0.87) = 9	ceiling(8 * 0.23) = 3
object x_2	object x_9	object x_3

FIGURE 3. Decoding example.

3.4.2. Objective function

After decoding a Y_i chromosome and defining the k medoids, each $(n-k)$ object of X must be allocated to its respective closest medoid. Thus, a partition associated with the Y_i chromosome will have been constructed. Then the silhouette index of this partition can be calculated [1] and corresponds to the objective function (*fitness*) to be maximized in the BRKGA. The objective is to find the chromosome mapped to the set of representative objects that produces the partition of X with the highest mean silhouette index value. In this work, the dissimilarity matrix D_{CO} is calculated from the co-association matrix CO (described in Sect. 2). This matrix presented interesting results when used with hierarchical algorithms [18]. Thus, the silhouette index is calculated using D_{CO} as the distance matrix.

$$D_{\text{CO}}(x_i, x_j) = 1 - \text{CO}(x_i, x_j),\ \forall\, i, j = 1, \ldots, n. \tag{2}$$

Silhouette index (Average Silhouette Width) [1,47]: takes values in the interval $[-1, 1]$ and considers cohesion and separation. Cohesion is measured by considering the distance between all objects in the same cluster, and the separation is based on the distances between objects from different clusters. The complexity of calculating this index is $\mathcal{O}(dn^2)$.

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}, i = 1, \ldots, n \tag{3}$$

where,

$$a(i) = \frac{1}{n_r - 1} \sum_{x_j \in C_r, i \neq j} D_{\text{CO}}(x_i, x_j),\ \forall\, x_i, x_j \in C_r,\ \forall\, C_r \in \pi,\ r = 1, \ldots, k$$

$$b(i) = \min_{C_u \in \pi \setminus C_r} \left\{ \frac{1}{n_u} \sum_{\forall x_j \in C_u} D_{\text{CO}}(x_i, x_j) \right\},\ \forall\, x_i \in C_r,\ \forall\, C_r, C_u \in \pi,\ r, u = 1, \ldots, k$$

where C_r and C_u are the rth and uth clusters, and n_r and n_u correspond, respectively, to the number of objects in clusters C_r and C_u.

$a(i)$ corresponds to the average distance between the object $x_i \in C_r$ and all the other objects of the same cluster. The smaller this value, the better the allocation of the object x_i to its cluster (in the sense of cohesion).

The average distance between the object $x_i \in C_r$ and all the objects of the same cluster is calculated for each cluster (other than C_r). $b(i)$ corresponds to the average distance between object x_i and all objects in the closest cluster.

The formula for calculating the average silhouette is presented below.

$$Sil(\pi) = \frac{1}{n}\sum_{i=1}^{n} s(i),\ i = 1, \ldots, n. \tag{4}$$

3.4.3. Genetic operators

The genetic operators of BRKGA-CE (phase 2) are those described in Section 2: selection, mutation, and crossover.

The population (of the first generation) is made up of randomly generated p chromosomes. Then, the genetic operators are applied to the population of the current generation to generate the population of the next generation for t generations. Finally, the chromosome that presented the highest objective function value (silhouette), decoded from the last generation, corresponds to the set of k representative objects that will be used as a starting point in phase 3 (iterative procedure) for the construction of the consensus partition.

3.5. Phase 3: consensus partition building

In this phase, only the best set of k medoids (the chromosome that presented the highest objective function value) produced by BRKGA is considered. More specifically, considering this set, an iterative procedure is applied to produce the consensus partition.

Initially, each of the k clusters $(C_1, \ldots, C_r, \ldots, C_k)$ is formed only by its respective medoid, while the other $(n-k)$ objects of X are not allocated to any cluster. At each iteration, an object from X (not yet allocated) is selected and allocated to one of the k clusters until all objects are allocated. The object x_i selected at each iteration is the one that presents the smallest distance (calculated from the matrix D_{CO}) to one of the k clusters. The distance between an object x_i and a cluster C_r , $r = 1, \ldots, k$, corresponds to the average of the distances between x_i and the objects of C_r. In this calculation, only the κ[9] objects from C_r closest to x_i are considered.

In each iteration, the distances of each object x_i (which has not yet been allocated to any cluster) are calculated for each of the clusters to find the allocation with the shortest distance (*i.e.*, the pair (x_i, C_r) with the smallest distance). This distance between x_i and C_r corresponds to the average distance between x_i and κ objects (closest to x_i) from C_r.

That is, for each object x_i (not allocated), the distance of x_i is calculated for each of the k clusters, and the smallest distance found (associated with x_i) is stored. Then, the object x_i (among all the unallocated objects) that presented the smallest average distance is allocated to the cluster closest to x_i. This process corresponds to one iteration of phase 3, and it occurs $(n-k)$ times to allocate all $(n-k)$ objects to clusters.

Next, the formula for calculating the distance between an object x_i (not allocated to any cluster) and a cluster C_r is presented. It should be remembered that the distances are calculated from the matrix D_{CO}.

$$D(x_i, C_r) = \sum_{w=1}^{\kappa} D_{\text{CO}}(x_i, x_{i,w}^r)/\kappa \tag{5}$$

where,

x_i: corresponds to the ith object of X that has not yet been allocated to any cluster.

$x_{i,w}^r$: corresponds to the wth closest object to x_i and belonging to the C_r cluster. That is, this object is among the κ objects closest to x_i that are allocated to C_r.

The allocation of $(n-k)$ objects in k clusters obtained at the end of this procedure, after $(n-k)$ iterations, corresponds to a partition of the n objects of X into k clusters, named consensus partition π^*.

Figure 4 illustrates the iterative process through an example with a dataset containing $n = 7$ objects to be divided into $k = 2$ clusters, assuming $\kappa = 10$ (in this case, all objects in the cluster will be considered in the

[9]The parameter κ corresponds to the number of objects within each cluster that are considered in the calculation of the distance. The larger the κ, the more the clusters tend to be compact (since all objects in a cluster must have a low distance between them). The clusters can be of elongated format for small values of κ because only the distance between the nearest objects is considered.

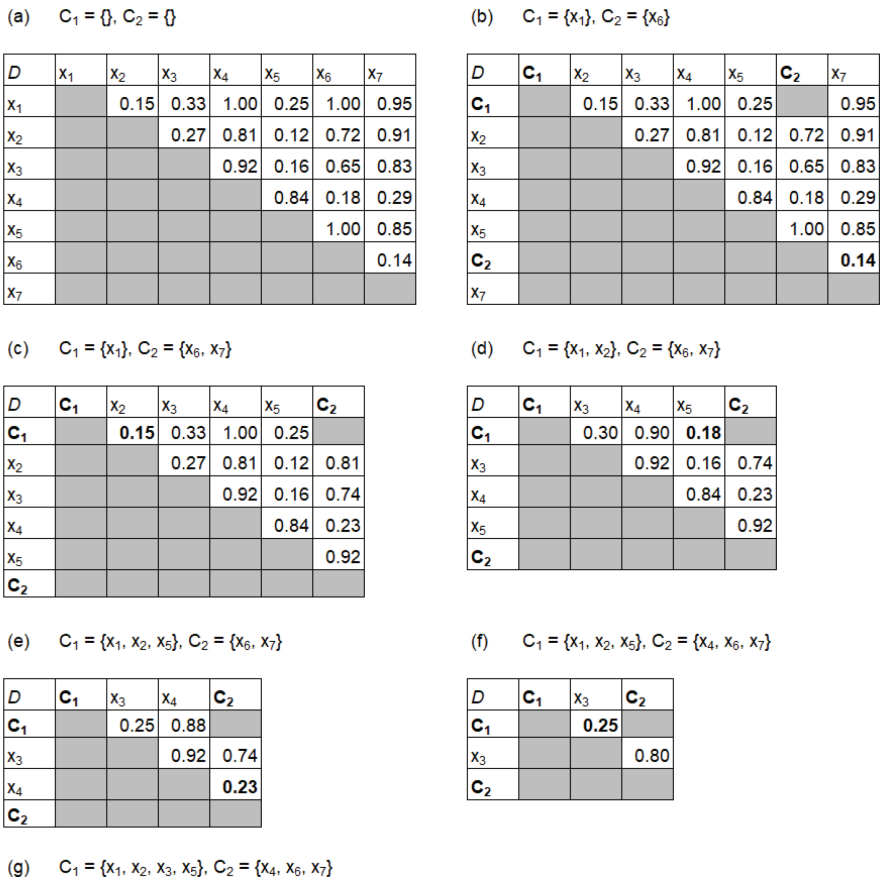

FIGURE 4. BRKGA-CE: third phase.

calculation, since no cluster has more than 10 objects). The objects x_1 and x_6 are the initial representative objects (best solution – phase 2). In the first iteration, each cluster C_r is composed of only the representative object λ_r, where $r \in \{1, 2\}$. At the end of the procedure, all n objects from X will be allocated to one of the k clusters. The matrix that shows the distances between objects and clusters has its values updated at each iteration.

The time complexity of the phase 3 is $\mathcal{O}((n-k)^2 * k * n_{\text{kmax}} * log(n_{\text{kmax}}))$, where n is the number of objects, k is the number of clusters (and the number of representative objects) and n_{kmax} is the number of objects in the cluster with the largest number of objects.

4. Computational experiments

This section presents the results and analyses obtained from the computational experiments carried out with the BRKGA-CE. All experiments were performed on a computer with an i5-6400 processor (2.70 GHz) and 16 GB RAM in the Windows 7 Operating System. The code and datasets used in this paper are online in a Github repository: https://github.com/aPizano/BRKGA-Cluster-Ensemble.

TABLE 2. Description of *benchmark* datasets.

Datasets	n	d	Classes	Source
Iris	150	3	3	UCI MLR
Wine	178	14	3	UCI MLR
Sonar	208	61	2	UCI MLR
Glass	214	11	6	UCI MLR
New-thyroid	215	6	3	UCI MLR
Seeds	218	8	3	UCI MLR
Flame	240	2	2	artificial data
Heart failure	299	13	2	UCI MLR
Pathbased	300	2	3	artificial data
Spiral	312	2	3	artificial data
Ecoli	336	9	8	UCI MLR
Ionosphere	351	35	2	UCI MLR
Libras	360	90	15	UCI MLR
Jain	373	2	2	artificial data
Compound	399	2	6	artificial data
WDBC	569	32	2	UCI MLR
Synthetic control	600	60	6	UCI MLR
R15	600	2	15	artificial data
Aggregation	788	2	7	artificial data
Yeast	1484	10	10	UCI MLR

The BRKGA-CE algorithm was compared to 8 algorithms in the literature,[10] including 19 variants (in parentheses), namely: BOM [15], EAC (EAC-SL, EAC-AL, EAC-CL) [18], SRS (SRS-SL, SRS-AL, SRS-CL) [31,32], CTS (CTS-SL, CTS-AL, CTS-CL) [31,32], ASRS (ASRS-SL, ASRS-AL, ASRS-CL) [31], WEAC (WEAC-SL, WEAC-AL, WEAC-CL) [25] LWEA (LWEA-SL, LWEA-AL, LWEA-CL) [26], and HGCEA [57]. The BRKGA-CE, BOM, EAC, WEAC and LWEA and all its variants were implemented in R by the authors of this article. All variants of the algorithms SRS, CTS and ASRS are implemented in the R package 'diceR' [11].

For the experiments of this work, 20 classification datasets[11] known in the literature were considered, with different characteristics (number of objects between 150 and 1484, attributes between 2 and 90, and classes between 2 and 15), as shown in the Table 2. The first column brings the name of the datasets, in the second column, the number of objects (n), in the third, the number of attributes (d), and in the fourth, the number of classes (the number of clusters k). Finally, the fifth column indicates the origin of each dataset: (i) artificial datasets[12] [16], and (ii) real-world datasets from *UCI Machine Learning Repository.*[13]

To assess the quality of the consensus partitions, two external validation indices (commonly used in other cluster ensemble works [26]) were considered, namely: the Adjusted Rand Index ($\mathcal{AR}$) [54] and Normalized Mutual Information ($\mathcal{NMI}$) [49]. In this case, a consensus partition π^* is compared with the true partition π^{gt} (considering the classes of objects as clusters), and a *score* is calculated that indicates the quality of π^*. Below are the equations for calculating the *scores*.

$$\mathcal{NMI}(\pi^*, \pi^{gt}) = \frac{\mathcal{I}(\pi^*, \pi^{gt})}{\sqrt{\mathcal{H}(\pi^*)\mathcal{H}(\pi^{gt})}}. \tag{6}$$

[10]The algorithms chosen for comparison are well known in the literature and have already been used for comparison in several articles [5,25,27,33,36,59,60].

[11]These datasets are well known in the literature and have been used in other cluster ensemble works cited in this paper.

[12]http://cs.uef.fi/sipu/datasets/.

[13]https://archive.ics.uci.edu/ml/index.php.

Where $\mathcal{I}(\pi^*, \pi^{gt})$ corresponds to the mutual information between the partitions π^* and π^{gt} and $\mathcal{H}(\pi)$ corresponding to the entropy of the partition π [49].

$$\mathcal{AR}(\pi^*, \pi^{gt}) = \frac{2(N_{00}N_{11} - N_{01}N_{10})}{N_a + N_b} \tag{7}$$

where,

$$N_a = (N_{00} + N_{01})(N_{01} + N_{11})$$

$$N_b = (N_{00} + N_{10})(N_{10} + N_{11}).$$

Where N_{11} is the number of pairs of objects belonging to the same cluster in π^* and π^{gt}. N_{00} is the number of pairs of objects belonging to different clusters in π^* and π^{gt}. N_{10} is the number of pairs of objects that belong to the same cluster in π^* and belong to different clusters in π^{gt}. N_{01} is the number of pairs of objects that belong to different clusters in π^* and belong to the same cluster in π^{gt} [54].

The parameters of BRKGA-CE were calibrated using irace (*Iterated racing for automatic algorithm configuration*) [40], available in the package 'irace' of R. Six datasets[14] were considered in the calibration experiments: Flame, Spiral, Jain, Iris, Ecoli, WDBC. Additionally, 1000 iterations were considered[15] of irace for the calibration of the five parameters and the evaluation of the quality of the solutions $\mathcal{NMI}$, which is often used in the literature [18, 26, 49]. The calibrated parameters were: p (population size), p_e (number of chromosomes from Y_e), p_m (number of mutant chromosomes), ρ (crossing probability), and κ (number of nearest neighbors considered in phase 3) and their respective values considered in the experiments were (BRKGA parameters as recommended in [22]): $p \in \{50, 75, 100\}$, $p_e \in \{0.1p, 0.2p, 0.25p\}$, $p_m \in \{0.1p, 0.2p, 0.3p\}$, $\rho \in \{0.6, 0.7, 0.8\}$, plus the parameter $\kappa \in \{10, 20, 30\}$. The number of generations was fixed at 200 since, in preliminary experiments, no improvement or significant contribution (gains in the objective function) was observed for more than 200 generations. From the results of experiments with irace, the best set of values found for the parameters was: $p = 50, p_e = 0.25p, p_m = 0.3p, \rho = 0.6, \kappa = 10$. The rounding rule used for p_e and p_m is to apply the ceiling function.

4.1. Results

The dataset partitions were generated by adopting, in the first phase of BRKGA-CE, the three strategies presented in Section 3.3. All executions were carried out with an ensemble size $m = 30$,[16] corresponding to the average of the values adopted in the literature in [18,26,27,31,32] (since values between 10 and 50 are adopted). To guarantee a fair comparison, the base partitions used are the same for all algorithms (BRKGA-CE and all algorithms from the literature).

To analyze the performance of the algorithms, inspired by an important work of literature [26], hypothesis tests were applied to validate the results, according to the Tables 3 and 4. Thus, considering the 20 datasets, the BRKGA-CE and the other algorithms in the literature were executed 10 times, and the values of $\mathcal{AR}$ and $\mathcal{NMI}$ obtained in 10 runs.

Thus, the *Wilcoxon* test was applied to verify whether the difference between the mean of BRKGA-CE results and a certain algorithm in the literature was significant for a given dataset, considering a significance level of 5%. If the difference is significant (p-value $\leq 5\%$) and the BRKGA-CE presents a better (or worse) result, then it is said that it is significantly better (or worse) for that dataset. Otherwise (p-value $> 5\%$), the algorithms

[14] Synthetic and real-world datasets with different number of objects, attributes, and classes.

[15] The authors suggest that the maximum number of iterations depends on the number of parameters, for example, one of the algorithms has 16 parameters and is calibrated with 5000 iterations, while another algorithm has 26 parameters and is calibrated with 10000 iterations [40].

[16] Each of the strategy's algorithms is sequentially executed (one after the other) choosing an integer value of k at random in the set $\{2, 3, \ldots, \sqrt{n}\}$. When all algorithms are executed, it goes back to the first algorithm to continue the generation of base partitions until the ensemble has a total of 30 partitions.

TABLE 3. Results of the hypothesis tests (*Wilcoxon* test) with a significance level of 5%). Counting the number of times the BRKGA-CE presents the following results: (Better/Equivalent/Worse). In bold the cases in which the BRKGA-CE presents a higher count of cases better than Equivalent and Worse.

	Strategy 1		Strategy 2		Strategy 3	
Algorithm	$\mathcal{AR}$	$\mathcal{NMI}$	$\mathcal{AR}$	$\mathcal{NMI}$	$\mathcal{AR}$	$\mathcal{NMI}$
BOM	(7/6/7)	(8/9/3)	**(10/3/7)**	**(10/5/5)**	(7/7/6)	**(9/5/6)**
EAC-SL	**(9/6/5)**	**(9/5/6)**	**(11/6/3)**	**(10/5/5)**	**(10/8/2)**	**(9/7/4)**
EAC-AL	(2/13/5)	(2/13/5)	**(10/8/2)**	(8/9/3)	(4/14/2)	(3/15/2)
EAC-CL	(7/12/1)	(8/10/2)	**(12/7/1)**	**(11/8/1)**	(8/11/1)	(8/10/2)
CTS-SL	(8/9/3)	(7/9/4)	**(11/7/2)**	**(10/6/4)**	**(9/9/2)**	**(9/8/3)**
CTS-AL	(1/14/5)	(3/12/5)	**(11/7/2)**	**(10/7/3)**	(5/13/2)	(5/13/2)
CTS-CL	(5/11/4)	(6/11/3)	**(12/7/1)**	**(12/7/1)**	(6/13/1)	(6/12/2)
SRS-SL	(8/8/4)	(7/8/5)	**(9/8/3)**	(7/7/6)	**(8/11/1)**	(7/9/4)
SRS-AL	(1/13/6)	(1/13/6)	(8/9/3)	(8/8/4)	(4/13/3)	(5/12/3)
SRS-CL	(6/11/3)	(7/10/3)	(9/10/1)	(7/11/2)	(4/15/1)	(6/12/2)
ASRS-SL	**(11/6/3)**	**(10/7/3)**	**(11/7/2)**	**(10/8/2)**	**(12/7/1)**	**(12/5/3)**
ASRS-AL	(5/8/7)	(5/12/3)	**(10/8/2)**	**(10/9/1)**	(3/14/3)	(4/13/3)
ASRS-CL	(4/11/5)	(6/11/3)	**(9/9/2)**	(8/10/2)	(5/14/1)	(6/12/2)
WEAC-SL	**(9/6/5)**	**(8/7/5)**	**(11/7/2)**	**(10/6/4)**	**(12/6/2)**	**(11/4/5)**
WEAC-AL	(2/14/4)	(5/12/3)	(8/10/2)	(7/11/2)	(3/13/4)	(4/12/4)
WEAC-CL	**(10/10/0)**	(9/10/1)	**(10/8/2)**	(8/10/2)	(6/12/2)	(6/12/2)
LWEA-SL	**(12/6/2)**	**(12/5/3)**	**(11/7/2)**	**(9/7/4)**	**(13/6/1)**	**(10/7/3)**
LWEA-AL	(2/12/6)	(2/14/4)	(6/12/2)	(5/12/3)	(3/16/1)	(5/13/2)
LWEA-CL	(9/10/1)	(9/9/2)	(8/11/1)	(6/12/2)	(5/13/2)	(8/10/2)

are said to have equivalent (or similar) performance. When compared with a given algorithm, each entry in the Table 3 indicates the count of results (Better/Equivalent/Worse) of the BRKGA-CE. The first line of this Table, for example, compares the results of the BRKGA-CE with the BOM algorithm, considering the $\mathcal{AR}$ and the $\mathcal{NMI}$ and the three strategies.

Comparing against the BOM algorithm considering, for example, $\mathcal{AR}$ and strategy 2, BRKGA-CE presented significantly better results in 10 datasets, significantly worse in 7 datasets, and results with no significant difference in 3 datasets.

From the Table 3, the best BRKGA-CE results are obtained in strategy 2, where the BRKGA-CE presents superior results than all 19 algorithms from the literature for the measures $\mathcal{AR}$ and $\mathcal{NMI}$. In the case of Strategy 3, BRKGA-CE outperforms 18 out of the 19 algorithms when considering the $\mathcal{NMI}$. In the case of $\mathcal{AR}$, BRKGA-CE outperforms 17 out of the 19 algorithms. The worst comparative results of BRKGA-CE were obtained in strategy 1, which includes only non-hierarchical algorithms.

In addition to the *Wilcoxon*, the *Friedman* [20] test was applied 6 times (2 measures $\times$ 3 strategies) to assess whether there is a significant difference between the average ranks[17] of the compared algorithms, considering a significance level of 5%. The values in the Table 4 indicate the average rank of each algorithm (in this case, the higher the average rank, the better). From the Table 4, it can be noticed that all the results are significant considering the p-value, which suggests that the difference between the compared algorithms is statistically significant in all cases.

[17]A rank of an algorithm corresponds to the value that indicates the position of that algorithm when ordered in front of the others, according to some measure of quality. The average rank of the algorithm corresponds to the average of the positions it occupies considering all the datasets used in the experiments.

TABLE 4. *Friedman* test results with a significance level of 5%. It presents the average ranks of each algorithm (considering the 20 datasets). All tests have 19 degrees of freedom. In bold the top five average ranks to facilitate the identification of the best results.

	Strategy 1		Strategy 2		Strategy 3	
	$\mathcal{AR}$	$\mathcal{NMI}$	$\mathcal{AR}$	$\mathcal{NMI}$	$\mathcal{AR}$	$\mathcal{NMI}$
χ_f^2	77.433	57.388	53.226	37.711	82.688	55.201
p-value	5.1E-09	9.9E-06	4.3E-05	6.4E-3	6.3E-10	2.1E-05
BRKGA-CE	12.20	**12.70**	**14.38**	**13.28**	**13.58**	**13.33**
BOM	10.55	8.40	**12.60**	11.10	12.50	10.30
EAC-SL	8.63	10.13	9.25	9.75	7.10	8.55
EAC-AL	**13.40**	**12.80**	9.43	9.43	11.43	**11.93**
EAC-CL	7.93	8.33	9.55	9.95	9.63	9.78
CTS-SL	9.30	10.80	7.33	8.13	6.05	7.25
CTS-AL	**14.05**	**13.40**	6.85	7.25	9.65	10.05
CTS-CL	11.00	10.15	8.65	9.25	10.38	9.98
SRS-SL	9.90	10.40	10.03	11.18	8.60	9.85
SRS-AL	**13.90**	**13.60**	**12.88**	**12.08**	**13.20**	**12.65**
SRS-CL	10.75	10.60	12.28	11.63	11.95	11.60
ASRS-SL	5.63	5.83	7.40	7.90	5.45	6.30
ASRS-AL	12.90	11.80	10.25	9.25	**12.70**	11.05
ASRS-CL	12.00	10.60	11.05	10.95	11.08	10.38
WEAC-SL	10.08	11.08	9.15	9.60	8.08	9.33
WEAC-AL	**13.15**	12.65	11.43	10.98	**14.08**	**14.58**
WEAC-CL	7.60	8.80	11.93	**12.68**	**12.70**	11.75
LWEA-SL	5.83	5.88	8.90	8.90	6.28	6.38
LWEA-AL	**13.65**	**13.20**	**13.33**	**13.28**	**13.70**	**13.55**
LWEA-CL	7.58	8.88	**13.38**	**13.48**	11.90	11.45

In the case of strategy 1, the BRKGA-CE presented, among the 20 algorithms, the 7th best result regarding the $\mathcal{AR}$ and the 4th best result regarding $\mathcal{NMI}$. In the case of strategy 2, BRKGA-CE presented, among the 20 algorithms, the best result regarding $\mathcal{AR}$ and the 2nd best result regarding $\mathcal{NMI}$. In the case of strategy 3, the BRKGA-CE presented, among the 20 algorithms, the 3rd best result regarding the $\mathcal{AR}$ and the $\mathcal{NMI}$.

From such results, it is observed that the BRKGA-CE, in general, is among the best results, considering several scenarios. In most comparisons, BRKGA-CE was among the top five and never worse than the 7th position compared to the other 19 algorithms.

Although the *Friedman* [20] test indicates a significant difference between the average ranks of the compared algorithms, it does not indicate which algorithms are responsible for such a difference. Thus, in order to complement the analysis, the *Wilcoxon* test was applied (significance level of 5%) in the 6 cases (3 strategies and 2 measures) to verify if there is a significant difference between the BRKGA-CE results and the algorithm with the best average rank for each of the strategies and evaluation indices, according to the Table 4.

From the application of this test, it was observed, in all cases, that there is no significant difference between the BRKGA-CE and the algorithms that presented the best average rank in any case. In the only case in which there was a significant difference (strategy 2, measure $\mathcal{AR}$), BRKGA-CE presented the best average rank and was compared to the algorithm with the second best average rank, the LWEA-CL.

Figure 5 illustrates the comparison between BRKGA-CE and other algorithms in the literature, according to the results of Table 4. The algorithms are ordered according to the $\mathcal{NMI}$ values in strategy 2.

In addition to the analyzes presented about the quality of the algorithms' solutions, processing times were also analyzed. In the Table 5 are presented median and the average processing times (in seconds) of each algorithm.

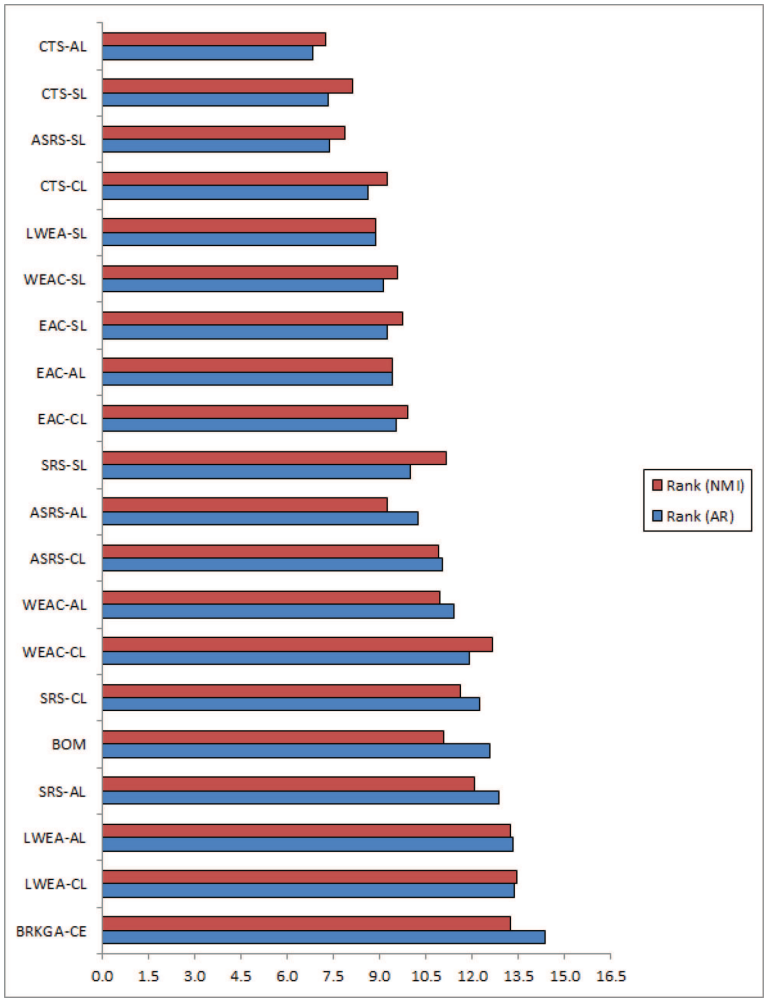

FIGURE 5. Results of strategy 2 in Table 4 – ordered by $\mathcal{AR}$ values.

By the Table 5, it is possible to observe the growth of the processing times. The algorithm with the shortest (best) mean times is the EAC, while the algorithm with the highest (worst) mean times is the BOM.
The processing time of the proposed method was longer than that of some methods in this aspect, mainly due to being based on a metaheuristic. BRKGA-CE is iterative and has computationally intensive procedures. Comparison with other algorithms is not suitable.

Figure 6 shows the BRKGA-CE processing times for each strategy. Outliers were removed to avoid distorting the figure, as some values exceed 2000 seconds.

As an example, results are visually represented for two-dimensional datasets produced by BRKGA-CE and LWEA-AL. These figures allow observing that, for some datasets, it is a difficult task to build clusters of non-compact format. Still in this sense, in relation to the figures, the presentation was restricted to the results of the BRKGA-CE and the LWEA-AL, since the LWEA-AL is the algorithm that presented the best results in the literature. Figure 7 presents the results in which both BRKGA-CE and LWEA-AL produce good quality solutions for the Flame dataset. One can perceive the quality of the solutions by the good identification and good definition of the clusters that is corroborated by high values of the validation indices ($\mathcal{NMI}$ and $\mathcal{AR}$).

TABLE 5. Processing times in seconds (average of 10 executions). Median and mean of each algorithm.

	Strategy 1		Strategy 2		Strategy 3	
Algorithm	Median	Mean	Median	Mean	Median	Mean
BRKGA-CE	18.6	85.9	14.6	81.7	17.6	79.4
BOM	6.4	231.6	5.9	223.1	6.8	219.8
EAC-SL	**0.1**	**0.4**	**0.1**	**0.4**	**0.1**	**0.4**
EAC-AL	**0.1**	**0.4**	**0.1**	**0.4**	**0.1**	**0.4**
EAC-CL	**0.1**	**0.4**	**0.1**	**0.4**	**0.1**	**0.4**
CTS-SL	5.4	7.9	5.3	8.4	5.4	7.7
CTS-AL	5.4	8.3	5.4	8.6	5.3	8.1
CTS-CL	5.4	8.3	5.3	8.2	5.3	7.9
SRS-SL	17.9	38.1	17.9	38.9	18.0	36.0
SRS-AL	18.8	37.3	18.3	37.4	17.9	36.2
SRS-CL	18.2	36.5	18.2	36.7	18.0	36.3
ASRS-SL	3.0	5.4	2.9	5.3	3.1	5.5
ASRS-AL	3.0	5.5	3.0	5.4	3.0	5.5
ASRS-CL	3.0	5.4	3.0	5.4	3.1	5.4
WEAC-SL	0.4	0.7	0.4	0.6	0.4	0.6
WEAC-AL	0.4	0.6	0.4	0.6	0.4	0.6
WEAC-CL	0.4	0.6	0.4	0.6	0.4	0.6
LWEA-SL	0.9	1.5	1.0	1.5	1.0	1.5
LWEA-AL	0.9	1.5	1.0	1.5	1.0	1.5
LWEA-CL	1.0	1.5	0.9	1.6	0.9	1.5

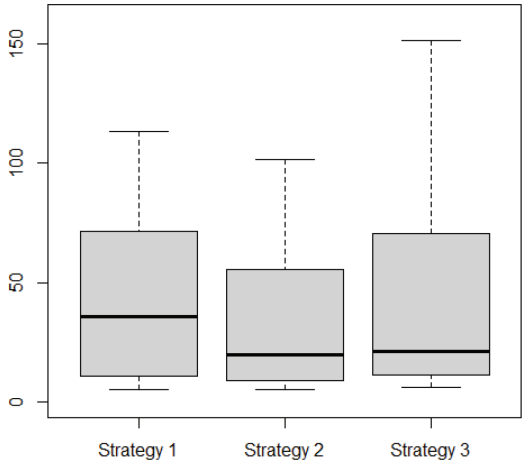

FIGURE 6. Processing times of BRKGA-CE. Boxplot without outliers.

Figure 8 presents the results in which both BRKGA-CE and LWE-AL produce lower quality solutions for the Spiral dataset. It can be seen that none of the algorithms was able to correctly separate the 3 clusters of spirals. This is possibly due to the difficulty of the dataset, which presents clusters in a spiral format.

Considering Strategy 2, BRKGA-CE achieved better results than LWE-AL (according to the average $\mathcal{AR}$ score) on the following datasets: Pathbased, Spiral, R15, Iris, Wine, Sonar, Glass, New-thyroid, Libras, Synthetic control, Yeast.

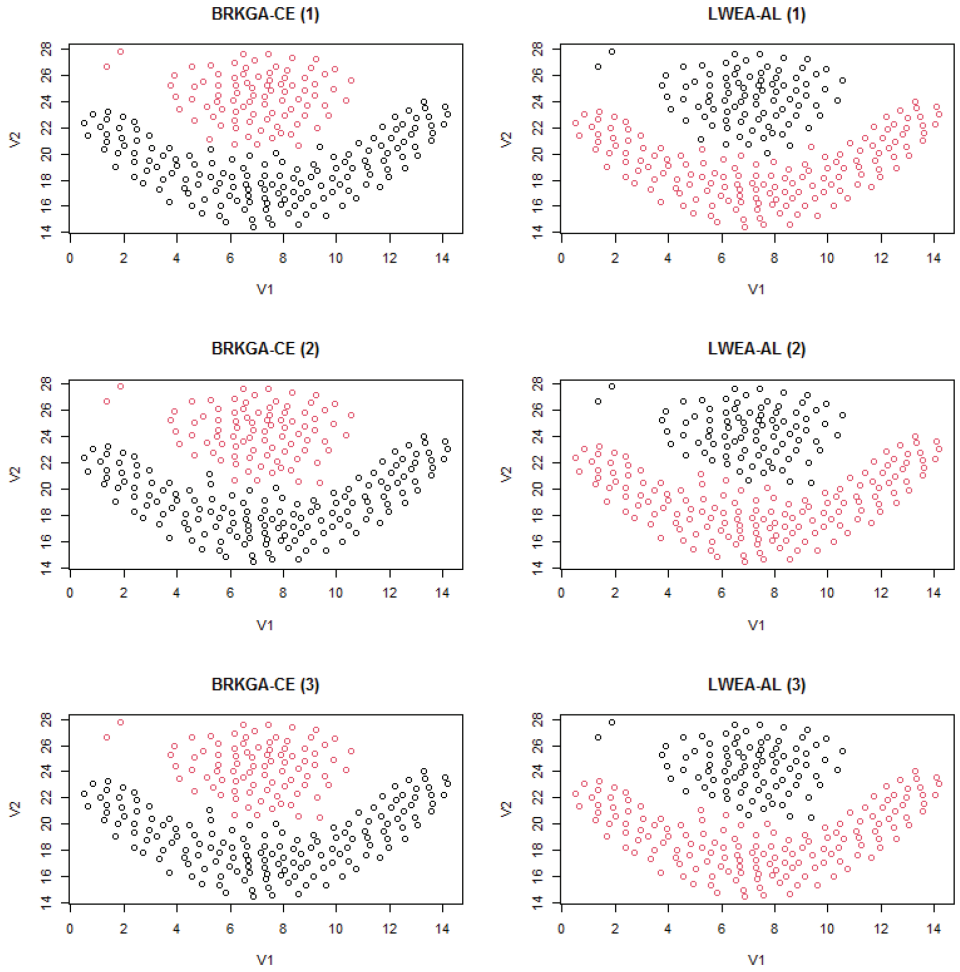

FIGURE 7. Solutions produced by BRKGA-CE and LWEA-AL for the Flame dataset. Strategies 1, 2 and 3.

TABLE 6. Comparison between BRKGA-CE consensus partitions and each base partition.

Strategy 1		Strategy 2		Strategy 3	
ARI	NMI	ARI	NMI	ARI	NMI
88.25%	98.15%	91.85%	99.35%	91.45%	99.80%

Continuing with algorithm analysis, Table 6 shows the percentage of counts in which BRKGA-CE presented a better solution when compared to each base partition considering all 20 datasets and 3 different strategies.

As a final experiment, BRKGA-CE was compared to a recent work in the literature that uses a genetic algorithm to solve the CEP [57].

Table 7 shows the comparison of BRKGA-CE with the results of HGCEA [57] and other algorithms from the literature presented in the paper. Base partitions used in this case were generated by k-means, k-medoids, fuzzy c-means (FCM), expectation maximization with Gaussian mixture (EMGM) and affinity propagation clustering

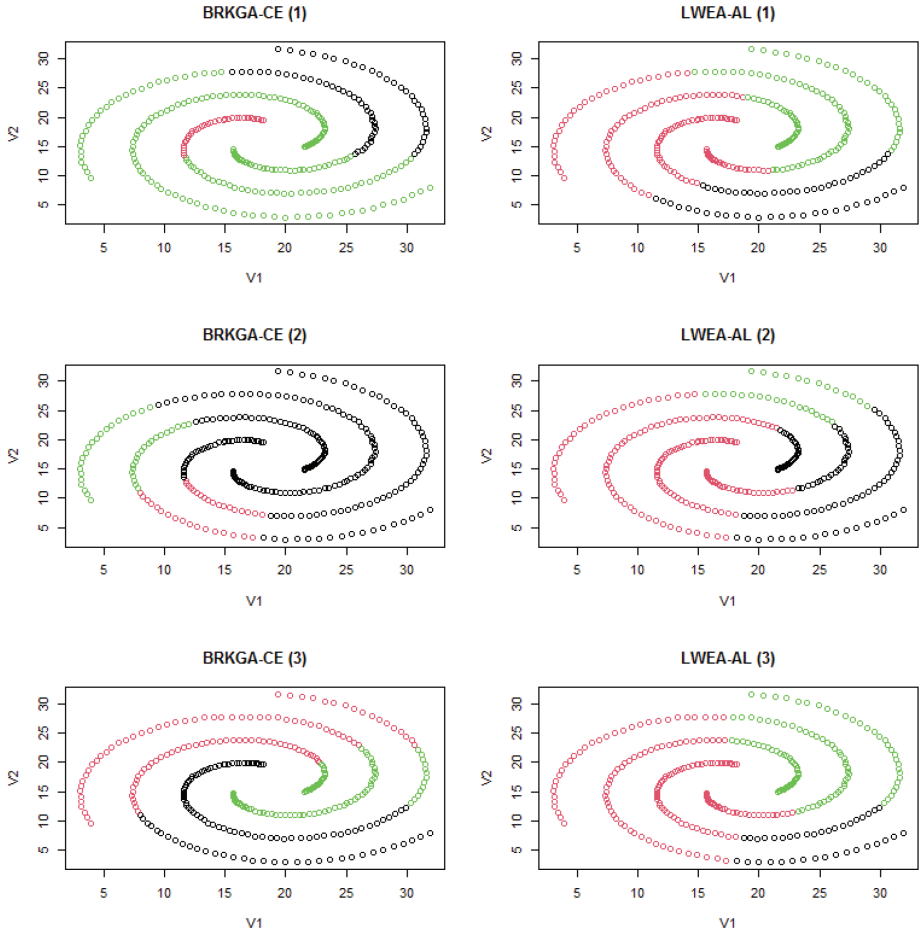

FIGURE 8. Solutions produced by BRKGA-CE and LWEA-AL for the spiral dataset. Strategies 1, 2 and 3.

TABLE 7. $\mathcal{NMI}$ Comparison between BRKGA-CE and the results of HGCEA.

Dataset	BRKGA-CE	HGCEA	MDEC-HC	MDEC-SC	CGE	USPEC
Wine (n = 178, d = 13)	**0.8595**	0.8303	0.7699	0.8209	0.3952	0.8237
Pima (n = 768, d = 8)	0.0568	**0.7361**	0.0023	0.0067	0.0010	0.0002
Brea (n = 699, d = 9)	**0.6931**	0.4598	0.4248	0.4568	0.2596	0.0183

algorithm (AP) according to the original work with ensemble size of 50. BRKGA-CE was run 20 times and Table 7 shows the average $\mathcal{NMI}$ of BRKGA-CE against the results presented in 7.[18]

Of the three datasets used, BRKGA-CE won in two of them. In the Pima dataset it lost only to HGCEA. For the Pima dataset, most algorithms achieved poor predictive performance (accuracy below 6% with the exception of HGCEA). A closer inspection revealed a substantial number of outliers across all eight attributes, with one feature containing up to 45 extreme values. Such anomalies may negatively affect the stability of

[18]Some datasets with no missing data (or with a small amount of missing data) were chosen for comparison.

TABLE 8. Desirable properties of consensus partition. Strategy 1, measure $\mathcal{AR}$.

Datasets	BRKGA-CE average score	Base partitions average score	Consistency ($\mathcal{AR}$) average similarity	Novelty count
Iris	**0.60**	0.43	0.46	0
Wine	**0.88**	0.49	0.14	0
Sonar	0.00	0.03	**0.21**	0
Glass	**0.72**	0.41	0.45	0
New-thyroid	0.16	0.16	**0.57**	0
Seeds	**0.49**	0.29	0.16	0
Flame	**0.85**	0.25	0.22	0
Heart	-0.01	0.02	**0.17**	0
Pathbased	**0.47**	0.33	0.44	0
Spiral	0.05	**0.06**	0.05	0
Ecoli	**0.68**	0.40	0.44	0
Ionosphere	0.18	0.15	**0.37**	0
Libras	0.29	0.21	**0.56**	0
Jain	**0.71**	0.16	0.24	0
Compound	**0.55**	0.42	0.48	0
WDBC	**0.35**	0.21	0.00	0
Synthetic	**0.98**	0.62	0.62	0
R15	0.64	0.50	**0.52**	0
Aggregation	**0.78**	0.48	0.52	0
Yeast	**0.15**	0.11	**0.15**	0

the CE algorithm, which relies on distribution estimation. Therefore, preprocessing techniques such as outlier removal or robust scaling may be necessary to improve performance.

Table 8 and Tables A.1–A.5 present comparisons between the consensus partitions and the base partitions, allowing us to assess whether the consensus partitions generated by BRKGA-CE exhibit the desired properties.[19,20] The first column of each table shows the dataset name. The second and third columns report the average scores ($\mathcal{AR}$, $\mathcal{NMI}$) of the consensus partitions and the base partitions, respectively. The fourth column shows the average similarity between the consensus partition and the base partitions, while the last column indicates the number of times the consensus partition is identical to any of the base partitions. Comparing the scores of BRKGA-CE with those of the base partitions, we observe that BRKGA-CE generally produces higher-quality solutions, indicating robustness. The fourth column shows that the consensus partitions are not always highly similar to the base partitions. In cases where the base partitions are of low quality, the consensus partition also shows lower consistency. Regarding novelty, the last column confirms that no consensus partition generated by BRKGA-CE is identical to any of its base partitions, demonstrating that the method can produce new solutions unattainable by individual algorithms. Although no experiments were specifically designed to evaluate stability, the datasets were used without any outlier removal, and BRKGA-CE consistently outperformed the base partitions across multiple datasets. This indicates that the consensus solutions are reasonably stable under varying conditions. Tables A.1–A.5 are presented in Appendix A.

[19]Desirable properties [18, 50, 51, 53] are: (i) robustness [53]: better average performance. than individual clustering algorithms; (ii) consistency [53]: the consensus partition should be similar to the base partitions, generally measured using an external validation index; (iii) novelty [53]: the ensemble tends to produce solutions not achievable by any single clustering algorithm; (iv) stability [50, 53]: reduced sensitivity to noise and outliers.

[20]The consensus partition should closely match the true partition (*ground truth*), which corresponds to the classes (labels) in classification datasets when available [18].

The worst results of BRKGA-CE were observed in the Sonar and Heart datasets. The worst results of the base partitions were also in these two datasets, with an average score consistently equal to or below 0.1. This indicates that the quality of the consensus partition depends on a minimum quality level of the base partitions.

5. Conclusions and future work

This work presents a new algorithm for the cluster ensemble. It is a genetic algorithm that seeks to maximize the average silhouette index, considering the co-association matrix (built from the base partitions of the ensemble) representing the solution through representative objects of each cluster.

Several experiments were conducted, with three different strategies for generating the ensemble, and used two indices of external validation commonly adopted in the literature: $\mathcal{AR}$ and $\mathcal{NMI}$.

The results of the experiments demonstrated the capacity of BRKGA-CE to produce good-quality solutions for different datasets from different ensembles. This fact is corroborated by the application of hypothesis tests that demonstrate the effectiveness of the proposed approach. Furthermore, the best comparative results were obtained when using strategies 2 and 3, indicating that the strategy used to generate the ensemble has a direct impact on the consensus partition. In strategy 2, there is no evidence that BRKGA-CE was outperformed by any of the 19 literature algorithms compared in the experiments. In the final experiment, BRKGA-CE was compared to 5 algorithms from the literature (according to the results of [57]) and presented competitive results. BRKGA-CE was better than all 5 algorithms in 2 datasets and presented the second best result in 1 dataset.

In future works, other indices in BRKGA-CE should be tested in addition to the average silhouette, such as Davies-Bouldin and Calinski-Harabasz [1]. Another important issue is to explore the use of different consensus matrices. In this work, only the co-association matrix was used in BRKGA-CE (the same used in EAC [18]), however, several algorithms in the literature use enhanced versions of the similarity matrix, such as CTS and SRS [31]. Another interesting direction for future work is focus on improving the handling of non-convex clusters. This could be achieved by incorporating algorithms specifically designed for non-convex structures, such as density-based or spectral clustering, into the ensemble. Adaptive strategies for selecting or weighting base partitions and hybrid schemes combining diverse algorithms with BRKGA could further enhance consensus quality and better capture complex cluster patterns in real-world datasets.

Acknowledgments

The authors thank the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) (Process 140044/2021-3) and PROPPI/UFF (FOPESQ 2021 public notice) for the financial support for the development of this work.

References

[1] O. Arbelaitz, I. Gurrutxaga, J. Muguerza, J.M. Pérez and I. Perona, An extensive comparative study of cluster validity indices. *Pattern Recognit.* **46** (2013) 243–256.

[2] R. Avogadri and G. Valentini, Fuzzy ensemble clustering based on random projections for DNA microarray data analysis. *Artif. Intell. Med.* **45** (2009)173–183.

[3] H.G. Ayad and M.S. Kamel, Cluster-based cumulative ensembles, in Multiple Classifier Systems: 6th International Workshop, MCS 2005, Seaside, CA, USA, June 13–15, 2005. Proceedings 6. Springer, Berlin (2005) 236–245.

[4] H.G. Ayad and M.S. Kamel, Cumulative voting consensus method for partitions with variable number of clusters. *IEEE Trans. Pattern Anal. Mach. Intell.* **30** (2008) 160–173.

[5] J. Azimi and X. Fern, Adaptive cluster ensemble selection, in Twenty-First International Joint Conference on Artificial Intelligence (2009).

[6] J. Azimi, M. Mohammadi, A. Movaghar and M. Analoui, Clustering ensembles using genetic algorithm, in CAMPS 2006 – International Workshop on Computer Architecture for Machine Perception and Sensing, Conference Proceedings (2006).

[7] L. Bai, J. Liang and F. Cao, A multiple k-means clustering ensemble algorithm to find nonlinearly separable clusters. *Inf. Fusion* **61** (2020) 36–47.

[8] T. Boongoen and N. Iam-On, Cluster ensembles: a survey of approaches with recent extensions and applications. *Comput. Sci. Rev.* **28** (2018) 1–25.

[9] C. Boulis and M. Ostendorf, Combining multiple clustering systems, in Knowledge Discovery in Databases: PKDD 2004: 8th European Conference on Principles and Practice of Knowledge Discovery in Databases, Pisa, Italy, September 20–24, 2004. Proceedings 8. Springer, Berlin (2004) 63–74.

[10] J.A.d.M. Brito, A.C. Fadel, G.S. Semaan and F. Montenegro, Heuristics applied to minimization of the maximum-diameter clustering problem. *IEEE Lat. Am. Trans.* **19** (2021) 652–659.

[11] D.S. Chiu and A. Talhouk, diceR: an R package for class discovery using an ensemble driven approach. *BMC Bioinform.* **19** (2018) 1–4.

[12] A.L. Coelho, E. Fernandes and K. Faceli, Inducing multi-objective clustering ensembles with genetic programming. *Neurocomputing* **74** (2010) 494–498.

[13] R.M. de Oliveira, A.A. Chaves and L.A.N. Lorena, A comparison of two hybrid methods for constrained clustering problems. *Appl. Soft Comput.* **54** (2017) 256–266.

[14] W. Fangfei, J. Qingshan, C. Lifei and H. Zhiling, Clustering ensemble based on the fuzzy KNN algorithm, in Proceedings – SNPD 2007: Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing (2007).

[15] V. Filkov and S. Skiena, Integrating microarray data by consensus clustering. *Int. J. Artif. Intell. Tools* **13** (2004) 863–880.

[16] P. Fränti and S. Sieranoja, K-means properties on six clustering benchmark datasets. *Appl. Intell.* **48** (2018) 4743–4759.

[17] A.L. Fred and A.K. Jain, Data clustering using evidence accumulation, in Vol. 4 of 2002 International Conference on Pattern Recognition. IEEE (2002) 276–280.

[18] A.L. Fred and A.K. Jain, Combining multiple clusterings using evidence accumulation. *IEEE Trans. Pattern Anal. Mach. Intell.* **27** (2005) 835–850.

[19] M.A. Ganaie, M. Hu, A. Malik, M. Tanveer and P. Suganthan, Ensemble deep learning: a review. *Eng. Appl. Artif. Intell.* **115** (2022) 105151.

[20] S. García, A. Fernández, J. Luengo and F. Herrera, Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: experimental analysis of power. *Inf. Sci.* **180** (2010) 2044–2064.

[21] R. Ghaemi, N. bin Sulaiman, H. Ibrahim and N. Mustapha, A review: accuracy optimization in clustering ensembles using genetic algorithms. *Artif. Intell. Rev.* **35** (2011) 287–318.

[22] J.F. Gonçalves and M.G. Resende, Biased random-key genetic algorithms for combinatorial optimization. *J. Heuristics* **17** (2011) 487–525.

[23] J.F. Gonçalves and M.G. Resende, Random-key genetic algorithms (2018).

[24] Y. Hong and S. Kwong, To combine steady-state genetic algorithm and ensemble learning for data clustering. *Pattern Recognit. Lett.* **29** (2008) 1416–1423.

[25] D. Huang, J.H. Lai and C.D. Wang, Combining multiple clusterings via crowd agreement estimation and multi-granularity link analysis. *Neurocomputing* **170** (2015) 240–250.

[26] D. Huang, C.D. Wang and J.H. Lai, Locally weighted ensemble clustering. *IEEE Trans. Cybern.* **48** (2018) 1460–1473.

[27] D. Huang, C.-D. Wang, H. Peng, J. Lai and C.-K. Kwoh, Enhanced ensemble clustering via fast propagation of cluster-wise similarities. *IEEE Trans. Syst. Man Cybern. Syst.* **51** (2018) 508–520.

[28] D. Huang, C.-D. Wang, J.-S. Wu, J.-H. Lai and C.-K. Kwoh, Ultra-scalable spectral clustering and ensemble clustering. *IEEE Trans. Knowl. Data Eng.* **32** (2019) 1212–1226.

[29] D. Huang, C.-D. Wang, J.-H. Lai and C.-K. Kwoh, Toward multidiversified ensemble clustering of high-dimensional data: from subspaces to metrics and beyond. *IEEE Trans. Cybern.* **52** (2021) 12231–12244.

[30] L. Huilan, J. Furong and X. Xiaobing, Combining multiple clusterings using information theory based genetic algorithm, in 2006 International Conference on Computational Intelligence and Security, ICCIAS 2006 (2006).

[31] N. Iam-On and T. Boongoen, Pairwise similarity for cluster ensemble problem: link-based and approximate approaches. Transactions on Large-Scale Data-and Knowledge-Centered Systems IX. Springer, Berlin (2013) 95–122.

[32] N. Iam-On, T. Boongoen and S. Garrett, Refining pairwise similarity matrix for cluster ensemble problem with cluster relations, in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Springer, Berlin (2008).

[33] N. Iam-On, T. Boongoen, S. Garrett and C. Price, A link-based approach to the cluster ensemble problem. *IEEE Trans. Pattern Anal. Mach. Intell.* **33** (2011) 2396–2409.

[34] A.K. Jain, Data clustering: 50 years beyond k-means. *Pattern Recognit. Lett.* **31** (2010) 651–666.

[35] Y. Jia, S. Tao, R. Wang and Y. Wang, Ensemble clustering via co-association matrix self-enhancement. *IEEE Trans. Neural Netw. Learn. Syst.* **35** (2023) 11168–11179.

[36] S. Kannan, S. Ramathilagam and P. Chung, Effective fuzzy c-means clustering algorithms for data clustering problems. *Expert Syst. Appl.* **39** (2012) 6292–6300.

[37] L. Kaufman and P.J. Rousseeuw, Finding Groups in Data: An Introduction to Cluster Analysis, in *Wiley Series in Probability and Mathematical Statistics*. Wiley, New York (1990).

[38] A. Lima, A. Lima, B. Nogueira, M. Santos and R.G. Pinheiro, A multi-population brkga for the automatic clustering problem, in 2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC). IEEE (2021) 368–373.

[39] M.A. Londe, L.S. Pessoa, C.E. Andrade and M.G. Resende, Biased random-key genetic algorithms: a review. *Eur. J. Oper. Res.* **321** (2025) 1–22.

[40] M. López-Ibáñez, J. Dubois-Lacoste, L.P. Cáceres, M. Birattari and T. Stützle, The irace package: iterated racing for automatic algorithm configuration. *Oper. Res. Perspect.* **3** (2016) 43–58.

[41] J. MacQueen, Some methods for classification and analysis of multivariate observations, in Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability (1967).

[42] R. Martí, P.M. Pardalos and M.G.C. Resende, editors. *Handbook of Heuristics*. Springer, Cham (2018).

[43] B.G. Mirkin, Mathematical Classification and Clustering. Kluwer Academic Press, Dordrecht (1996).

[44] R.T. Ng and J. Han, CLARANS: a method for clustering objects for spatial data mining. *IEEE Trans. Knowl. Data Eng.* **14** (2002) 1003–1016.

[45] N. Nguyen and R. Caruana, Consensus clusterings, in Proceedings – IEEE International Conference on Data Mining, ICDM (2007).

[46] M. Resende, Introdução aos algoritmos genéticos de chaves aleatórias viciadas. Simpósio Brasileiro de Pesquisa Operacional (2011) 3680–3691.

[47] P.J. Rousseeuw, Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.* **20** (1987) 53–65.

[48] G.S. Semaan, A.C. Fadel, J.A.d.M. Brito and L.S. Ochi, A hybrid heuristic with hopkins statistic for the automatic clustering problem. *IEEE Lat. Am. Trans.* **17** (2019) 7–17.

[49] A. Strehl and J. Ghosh, Cluster ensembles – a knowledge reuse framework for combining multiple partitions. *J. Mach. Learn. Res.* **3** (2002) 583–617.

[50] A. Topchy, A.K. Jain and W. Punch, A mixture model for clustering ensembles in Proceedings of the 2004 SIAM International Conference on Data Mining. SIAM (2004) 379–390.

[51] A.P. Topchy, M.H. Law, A.K. Jain and A.L. Fred, Analysis of consensus partition in cluster ensemble, in Fourth IEEE International Conference on Data Mining (ICDM'04). IEEE (2004) 225–232.

[52] A. Topchy, A.K. Jain and W. Punch, Clustering ensembles: models of consensus and weak partitions. *IEEE Trans. Pattern Anal. Mach. Intell.* **27** (2005) 1866–1881.

[53] S. Vega-Pons and J. Ruiz-Shulcloper, A survey of clustering ensemble algorithms. *Int. J. Pattern Recognit. Artif. Intell.* **25** (2011) 337–372.

[54] N.X. Vinh, J. Epps and J. Bailey, Information theoretic measures for clusterings comparison: variants, properties, normalization and correction for chance. *J. Mach. Learn. Res.* **11** (2010) 2837–2854.

[55] L. Xu and S. Ding, Dual-granularity weighted ensemble clustering. *Knowl. Based Syst.* **225** (2021) 107124.

[56] R. Xu and D. Wunsch, Survey of clustering algorithms. *IEEE Trans. Neural Netw.* **16** (2005) 645–678.

[57] W. Yang, Y. Zhang, H. Wang, P. Deng and T. Li, Hybrid genetic model for clustering ensemble. *Knowl. Based Syst.* **231** (2021) 107457.

[58] Z. Yu, H.S. Wong and H. Wang, Graph-based consensus clustering for class discovery from gene expression data. *Bioinformatics* **23** (2007) 2888–2896.

[59] X. Zhang, L. Jiao, F. Liu, L. Bo and M. Gong, Spectral clustering ensemble applied to sar image segmentation. *IEEE Trans. Geosci. Remote Sens.* **46** (2008) 2126–2136.

[60] L. Zheng, T. Li and C. Ding, Hierarchical ensemble clustering, in 2010 IEEE International Conference on Data Mining. IEEE (2010) 1199–1204.

[61] P. Zhou, L. Du, Y.-D. Shen and X. Li, Tri-level robust clustering ensemble with multiple graph learning, in Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 35 (2021) 11125–11133.

[62] P. Zhou, X. Wang, L. Du and X. Li, Clustering ensemble via structured hypergraph learning. *Inf. Fusion* **78** (2022) 171–179.
[63] P. Zhou, X. Liu, L. Du and X. Li, Self-paced adaptive bipartite graph learning for consensus clustering. *ACM Trans. Knowl. Discov. Data* **17** (2023) 1–35.

Appendix A. Desirable properties of consensus partition

Tables A.1–A.5 present comparisons between the consensus partitions and the base partitions, allowing us to assess whether the consensus partitions generated by BRKGA-CE exhibit the desired properties.

Table A.1. Desirable properties of consensus partition. Strategy 1, measure $\mathcal{NMI}$.

Datasets	BRKGA-CE average score	Base partitions average score	Consistency ($\mathcal{NMI}$) average similarity	Novelty count
Iris	**0.69**	0.61	0.68	0
Wine	**0.86**	0.64	0.29	0
Sonar	0.00	0.10	**0.41**	0
Glass	**0.70**	0.57	0.65	0
New-thyroid	0.32	0.33	**0.69**	0
Seeds	**0.51**	0.48	0.41	0
Flame	**0.81**	0.49	0.46	0
Heart	0.00	0.05	**0.31**	0
Pathbased	0.56	0.52	**0.64**	0
Spiral	0.11	0.18	**0.29**	0
Ecoli	0.68	0.59	**0.69**	0
Ionosphere	0.17	0.24	**0.53**	0
Libras	0.58	0.46	**0.71**	0
Jain	**0.71**	0.47	0.53	0
Compound	**0.72**	0.67	**0.72**	0
WDBC	0.34	**0.40**	0.02	0
Synthetic	**0.99**	0.84	0.84	0
R15	**0.81**	0.72	0.75	0
Aggregation	**0.89**	0.78	0.78	0
Yeast	0.29	0.27	**0.49**	0

TABLE A.2. Desirable properties of consensus partition. Strategy 2, measure $\mathcal{AR}$.

Datasets	BRKGA-CE average score	Base partitions average score	Consistency ($\mathcal{AR}$) average similarity	Novelty count
Iris	0.57	0.47	**0.58**	0
Wine	**0.78**	0.33	0.23	0
Sonar	0.00	0.00	**0.28**	0
Glass	**0.66**	0.37	0.36	0
New-thyroid	0.17	0.09	**0.47**	0
Seeds	**0.56**	0.43	0.29	0
Flame	**0.88**	0.26	0.25	0
Heart	0.02	0.05	**0.29**	0
Pathbased	0.50	0.38	**0.52**	0
Spiral	0.03	**0.17**	0.07	0
Ecoli	**0.76**	0.32	0.31	0
Ionosphere	0.00	0.10	**0.21**	0
Libras	0.32	0.13	**0.38**	0
Jain	**0.81**	0.29	0.31	0
Compound	0.60	0.61	**0.66**	0
WDBC	0.01	**0.21**	0.16	0
Synthetic	**0.99**	0.61	0.60	0
R15	**0.62**	0.43	0.52	0
Aggregation	**0.87**	0.51	0.51	0
Yeast	**0.20**	0.07	0.18	0

TABLE A.3. Desirable properties of consensus partition. Strategy 2, measure $\mathcal{NMI}$.

Datasets	BRKGA-CE average score	Base partitions average score	Consistency ($\mathcal{NMI}$) average similarity	Novelty count
Iris	0.66	0.62	**0.72**	0
Wine	**0.78**	0.42	0.35	0
Sonar	0.02	0.07	**0.34**	0
Glass	**0.68**	0.50	0.52	0
New-thyroid	0.35	0.24	**0.56**	0
Seeds	**0.55**	0.44	0.45	0
Flame	**0.86**	0.47	0.45	0
Heart	0.06	0.07	**0.38**	0
Pathbased	0.57	0.52	**0.65**	0
Spiral	0.05	0.29	**0.32**	0
Ecoli	**0.72**	0.46	0.53	0
Ionosphere	0.03	0.14	**0.30**	0
Libras	**0.60**	0.35	0.56	0
Jain	**0.80**	0.53	0.55	0
Compound	0.78	0.74	**0.79**	0
WDBC	0.03	0.25	**0.27**	0
Synthetic	**0.99**	0.83	0.84	0
R15	**0.80**	0.63	0.70	0
Aggregation	**0.92**	0.79	0.79	0
Yeast	0.29	0.19	**0.40**	0

TABLE A.4. Desirable properties of consensus partition. Strategy 3, measure $\mathcal{AR}$.

Datasets	BRKGA-CE average score	Base partitions average score	Consistency ($\mathcal{AR}$) average similarity	Novelty count
Iris	**0.57**	0.45	0.49	0
Wine	**0.84**	0.38	0.18	0
Sonar	0.00	0.02	**0.19**	0
Glass	**0.77**	0.39	0.40	0
New-thyroid	0.17	0.12	**0.32**	0
Seeds	**0.58**	0.37	0.29	0
Flame	**0.90**	0.23	0.23	0
Heart	0.02	0.03	**0.17**	0
Pathbased	0.48	0.36	**0.49**	0
Spiral	**0.67**	0.36	0.44	0
Ecoli	0.09	0.12	**0.27**	0
Ionosphere	0.31	0.16	**0.42**	0
Libras	**0.81**	0.23	0.27	0
Jain	**0.71**	0.53	0.57	0
Compound	0.05	**0.22**	0.08	0
WDBC	**0.99**	0.65	0.65	0
Synthetic	**0.63**	0.46	0.50	0
R15	**0.85**	0.51	0.52	0
Aggregation	0.18	0.09	**0.19**	0
Yeast	0.05	**0.22**	0.08	0

TABLE A.5. Desirable properties of consensus partition. Strategy 3, measure $\mathcal{NMI}$.

Datasets	BRKGA-CE average score	Base partitions average score	Consistency ($\mathcal{NMI}$) average similarity	Novelty count
Iris	0.67	0.62	**0.68**	0
Wine	**0.83**	0.51	0.33	0
Sonar	0.02	0.09	**0.31**	0
Glass	**0.74**	0.53	0.57	0
New-thyroid	0.33	0.27	**0.49**	0
Seeds	**0.57**	0.45	0.43	0
Flame	**0.87**	0.47	0.46	0
Heart	0.06	0.06	**0.25**	0
Pathbased	0.56	0.51	**0.64**	0
Spiral	0.12	0.23	**0.37**	0
Ecoli	**0.68**	0.51	0.63	0
Ionosphere	0.08	0.18	**0.35**	0
Libras	**0.60**	0.39	**0.60**	0
Jain	**0.80**	0.50	0.52	0
Compound	**0.78**	0.72	0.75	0
WDBC	0.07	**0.32**	0.16	0
Synthetic	**0.99**	0.85	0.86	0
R15	**0.81**	0.67	0.72	0
Aggregation	**0.92**	0.79	0.79	0
Yeast	0.30	0.22	**0.44**	0