A Novel Grouping Harmony Search Algorithm for Clustering Problems

Itziar Landa-Torres[1], Diana Manjarres[1], Sergio Gil-López[1], Javier Del Ser[1,2,3], and Sancho Salcedo-Sanz[4]

[1] TECNALIA, E-48160 Derio, Spain,
{itziar.landa,diana.manjarres,sergio.gil,javier.delser}@tecnalia.com
[2] University of the Basque Country UPV/EHU, 48013 Bilbao, Spain
{javier.delser}@ehu.eus
[3] Basque Center for Applied Mathematics (BCAM), 48009 Bilbao, Spain
[4] Universidad de Alcalá, E-28871 Alcalá de Henares, Spain,
sancho.salcedo@uah.es

Abstract The problem of partitioning a data set into disjoint groups or clusters of related items plays a key role in data analytics, in particular when the information retrieval becomes crucial for further data analysis. In this context, clustering approaches aim at obtaining a good partition of the data based on multiple criteria. One of the most challenging aspects of clustering techniques is the inference of the optimal number of clusters. In this regard, a number of clustering methods from the literature assume that the number of clusters is known a priori and subsequently assign instances to clusters based on distance, density or any other criterion. This paper proposes to override any prior assumption on the number of clusters or groups in the data at hand by hybridizing the grouping encoding strategy and the Harmony Search (HS) algorithm. The resulting hybrid approach optimally infers the number of clusters by means of the tailored design of the HS operators, which estimates this important structural clustering parameter as an implicit byproduct of the instance-to-cluster mapping performed by the algorithm. Apart from inferring the optimal number of clusters, simulation results verify that the proposed scheme achieves a better performance than other naïve clustering techniques in synthetic scenarios and widely known data repositories.

Keywords: Clustering, Grouping Encoding, Harmony Search

1 Introduction

Clustering is an important subgroup of unsupervised learning technique that involves grouping data objects into groups or clusters, which may be disjoint (*crisp* clustering) or overlap among each other (*fuzzy* clustering) [1]. A loose definition of clustering could be casted as the process of classifying unlabeled objects into groups in such a way that the members within a determinate cluster (or group) are similar to each other under a given measure of similarity. In essence, clustering aims at grouping an input set of *samples* into a finite number of clusters using only the information contained in such samples. The so-called *samples* (also denoted in the related literature as observations or instances) are normally modeled as numerical vectors whose items (features) represent numerical information to be used in the similarity measure. Mathematically, given a feature space $\mathcal{U}$, and if $\boldsymbol{\mathcal{X}} \doteq \{\mathbf{X}_1, \ldots, \mathbf{X}_N\}$ denotes a set of N samples in such an space, the challenge of clustering problems lies in finding an K-sized optimal partition of $\boldsymbol{\mathcal{X}}$, i.e. $\boldsymbol{\mathcal{X}}^* = \{\boldsymbol{\mathcal{X}}_1^*, \ldots, \boldsymbol{\mathcal{X}}_K^*\}$ (with $\boldsymbol{\mathcal{X}}_k^* \bigcap \boldsymbol{\mathcal{X}}_{k'}^* = \emptyset \; \forall k \neq k'$, $\bigcup_{k=1}^K \boldsymbol{\mathcal{X}}_k^* = \boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{X}}_k^*$ denoting the k-th partition of $\boldsymbol{\mathcal{X}}^*$). This clustering arrangement collects in the same cluster samples that are similar to each other as measured by a given objective function $f(\boldsymbol{\mathcal{X}}^*)$, which can defined under different similarity-based criteria.

Most of the clustering techniques in the literature can be divided into two general classes: hierarchical [2] and partitional [3]. The first class corresponds to those methods that create a hierarchical decomposition of the dataset under study. They can be agglomerative, when they start the clustering process with each sample on a separate cluster and successively combine clusters; or divisive, if they begin with all the patterns in a single cluster and perform this partition. By contrast, partitional algorithms obtain a single partition of the data instead of a hierarchy, i.e. they begin with a initial partition that is iteratively refined in order to obtain the final solution.

Interestingly under the scope of this manuscript, clustering algorithms can be also sorted depending on the deterministic or stochastic nature of the underlying algorithm: as such, deterministic approaches are not controlled by any probabilistic process, hence the instance-to-cluster mapping is fixed whenever the parameters of the algorithm and the dataset being clustered do not vary along time. On the other hand, stochastic clustering models are governed by probability-based processes so that this randomness may help the search process escape from local optima, at the cost of a certain degree of variability imposed on the instance-to-cluster mapping even when the input data does not vary. It is also worth mentioning other clustering techniques that rely on other criteria with stochastic and deterministic ingredients in their algorithmic core, such as Tabu Search [4], neural networks [5] and kernel spaces [6].

In this paper we focus on stochastic clustering models ruled by meta-heuristic solvers, in particular those incorporating the so-called Harmony Search (HS) algorithm as their constituent heuristic engine. The authors in [7] present a framework for simultaneous feature selection and clustering using the HS algorithm, whereas in [8] a centralized cluster-based protocol based on HS is proposed to minimize the intra-cluster distance and thus optimize the energy consumption of wireless networks. The performance of HS is compared to that of conventional clustering techniques in [9]. Despite the massive upsurge of different clustering techniques along the years, to the authors' knowledge grouping-encoded algorithms have not been tested in clustering problems. Intuitively, the grouping algorithm should perform well when applied to clustering problems, since it is originally conceived and well-adapted to manage groups of items [10]. This work takes a step further by adapting a grouping-encoded HS to accommodate a varying number of groups (clusters), which is estimated along the search process undertaken by the HS operators. This novel ingredient is deemed of paramount importance for practical clustering scenarios where traditional tools in this regard (e.g. the Elbow method) have been proven not to be efficient nor effective in most cases.

2 Proposed Approach

The Grouping Harmony Search Algorithm for Clustering (GHSC) proposed in this paper adopts the classical grouping encoding first contributed by Falkenauer in [10] and recently implemented in [12]. This variable-length grouping encoding is carried out by splitting each candidate vector $\mathbf{s}$ handled by the heuristic solver into two parts, i.e. $\mathbf{s} = [\mathbf{s_x}|\mathbf{s_y}]$: the first part, $\mathbf{s_x}$, is the *assignment part*, which consists of N integer indices with values drawn from the set $\{1,\ldots,k_y\}$ with k_y denoting the length of the second part of the solution. This part establishes to which cluster is assigned each sample. The second part $\mathbf{s_y}$ of the encoded solution (*group part*) maintains a list of tags associated to each of the clusters of the solution. It is composed by a k_y-length vector of integer indices from the set $\{1,\ldots,K\}$, which serves as a indexing reference for the assignment part. Therefore, the length of $\mathbf{s_x}$ is fixed and equal to N for a given problem, whereas the length of the group part is not fixed and may vary among the solutions handled by the search process. The underlying heuristic solver searches for the best length $1 \leq k_y \leq K$ of the group part in terms of an objective function. Following this notation, a solution for a clustering problem with $N = 10$ samples and $k_y = 4$ clusters could be represented as [3 2 2 4 1 3 1 2 3 4 | 1 2 3 4] in which, according to the notation introduced in Section 1, the partition $\boldsymbol{\mathcal{X}}^*$ would be $\boldsymbol{\mathcal{X}}_1^* = \{\mathbf{X}_5, \mathbf{X}_7\}$, $\boldsymbol{\mathcal{X}}_2^* = \{\mathbf{X}_2, \mathbf{X}_3, \mathbf{X}_8\}$, $\boldsymbol{\mathcal{X}}_3^* = \{\mathbf{X}_1, \mathbf{X}_6, \mathbf{X}_9\}$ and $\boldsymbol{\mathcal{X}}_3^* = \{\mathbf{X}_4, \mathbf{X}_{10}\}$.

2.1 Redundancy of the Clustering Encoding

As often stated in related contributions, most solution encoding strategies proposed to represent groups suffer from redundancies [13,14]. This section will elaborate on how to properly design a cluster encoding that mitigates this redundancy by ensuring that a clustering arrangement cannot be represented by more than one encoded individual. To this end, in the following set of encoded solutions (with colors compounding the group part for clarity),

$$\begin{array}{c}
[1\ 2\ 3\ 2\ 3\ 1\ 2\ 1\ 3 \mid \mathrm{G\ R\ Y}],\\
[1\ 3\ 2\ 3\ 2\ 1\ 3\ 1\ 2 \mid \mathrm{G\ Y\ R}],\\
[3\ 1\ 2\ 1\ 2\ 3\ 1\ 3\ 2 \mid \mathrm{R\ Y\ G}],\\
[2\ 1\ 3\ 1\ 3\ 2\ 1\ 2\ 3 \mid \mathrm{R\ G\ Y}],\\
[2\ 3\ 1\ 3\ 1\ 2\ 3\ 2\ 1 \mid \mathrm{Y\ G\ R}],\\
[3\ 2\ 1\ 2\ 1\ 3\ 2\ 3\ 1 \mid \mathrm{Y\ R\ G}],
\end{array} \tag{1}$$

it can be noted that all the represented clustering arrangements are equal: elements $\{\mathbf{X}_1, \mathbf{X}_6, \mathbf{X}_8\}$ always belong to cluster G (green), $\{\mathbf{X}_2, \mathbf{X}_4, \mathbf{X}_7\}$ to cluster

R (red) and $\{\mathbf{X}_3, \mathbf{X}_5, \mathbf{X}_9\}$ to Y (yellow). The proposal made recently by the authors in [15] proposes to sort the group part according to a pre-established criterion. In this particular case, for example, the colors can be listed according to its wavelength λ ($\lambda_R = 618$, $\lambda_G = 497$ and $\lambda_Y = 570$ nanometers). As a result, the 6 individuals would be encoded as [1 3 2 3 2 1 3 1 2 | G Y R].

This being said, the proposal of this work is to sort the indices of the clusters in order of appearance along the assignment part, in such a way that the 6 equivalent solutions in (1) are encoded as [1 2 3 2 3 1 2 1 3 | 1 2 3]. This encoding reduces the solution space of the problem, as it implies that the first position in the solution vector will always be a 1 and higher indexes will be less likely to appear. Additionally, the group part of the harmony is not needed, as all the information from the solution can be extracted from the assignment part; it determines each of the samples to which cluster is assigned and the number of clusters k_y can be deduced from the maximum value in the assignment part, i.e. $k_y = \max \mathbf{s}_{\mathbf{x}}$. As a counterpoint the proposed solution encoding requires several modifications in the improvisation operators of the nominal Harmony Search algorithm, which is the heuristic engine selected to evolve this numerical representation of clustering arrangements.

2.2 Proposed Grouping Harmony Search Algorithm for Clustering

The heuristic solver that lies at the core of the proposed clustering scheme is Harmony Search (HS), a population-based meta-heuristic algorithm that since its invention in [11] has rendered excellent results in the field of combinatorial optimization [16]. It mimics the behavior of a music orchestra when aiming at composing the most harmonious melody, as measured by aesthetic standards. Just like jazz musicians improvise harmonies time after time searching for aesthetically pleasant melodies, the HS algorithm improves the fitness of the solution vector in an iterative fashion by applying several operators to a φ-sized set of solutions, stored in the so-called Harmony Memory (HM). The flow diagram of the HS algorithm can be summarized in four steps: (i) initialization of the HM; (ii) improvisation of a new harmony; (iii) update of the HM with the new generated harmony if its fitness improves that of the worst currently in the HM; and (iv) repeat termination criterion (e.g. maximum number of iterations or fitness stall) is satisfied. The improvisation procedure is controlled by two different probabilistic operators, which are sequentially applied to each variable so as to produce new improvised candidate solutions:

– The Harmony Memory Considering Rate, HMCR $\in [0, 1]$, establishes the probability that the new value for a certain variable is drawn uniformly from the values of this same note in all the remaining melodies. Otherwise (i.e. with a probability $1 - \text{HMCR}$), the value is chosen uniformly at random from the alphabet of the variable. This latter case is commonly referred to as *random consideration*. However, some works (e.g. [17]) implement the random consideration as a third, separated probabilistic operator.

– The Pitch Adjusting Rate, PAR $\in [0,1]$, sets the probability that the new value x_{new} for a given variable is drawn from its neighboring values in the alphabet. To yield a PAR operator well-suited to the problem at hand, the variable alphabet should be sorted according to the fitness to be optimized so that subtle pitch adjustments do not imprint large changes in the value of the fitness function.

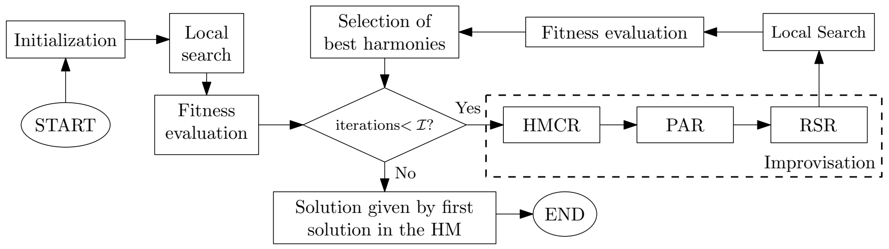

Figure 1. General scheme of the proposed GHSC algorithm.

The flow diagram of the proposed GHSC algorithm is schematically shown in Figure 1, and comprises 5 different steps:

A. Initialization, only executed at the first iteration: each of the φ candidate solutions included in the HM is assigned a random value from the set $\{1,\ldots,K\}$ (number of clusters), from which the N entries of the $\mathbf{s_x}$ part of the harmony are drawn uniformly at random.

B. Improvisation, which generates new harmonies by operating on the s_x part of each solution. This process is sequentially applied to each entry of the s_x part of every harmony in the HM. Additionally, as done in [12,15,17], the proposed improvisation procedure differs from the original HS implementation by introducing a third parameter *Random Selection Rate*, RSR $\in [0,1]$, which allows for an improved control of the tradeoff between the explorative and the exploitative behavior of the algorithm. Thus, the improvisation of the proposed GHSC is controlled by means of the HMCR, PAR and RSR operators. In light of the results obtained in previous works [17], the parameters of the CHS are modified so as to achieve a more effective information exchange and exploit information related to the distances among samples:

– HMCR: the proposed method does not exchange information between harmonies, but it leverages the information of the current partition $\boldsymbol{\mathcal{X}}^*$ and attempts to enhance it based on the proximity between samples. The HMCR applied to a certain sample establishes the probability that such an instance is assigned to another cluster by addressing the connections of the surrounding samples. These are the main steps:
  1. All the samples in the network are listed in increasing order of the distances to the sample to which the HMCR is applied (from smallest to largest). The first samples of this list are denoted as *nearby candidates*.

2. One out of the *nearby candidates* computed in the previous step is uniformly chosen at random, and the sample on which HMCR is applied is assigned to the cluster of the chosen *nearby candidate*.

– PAR: as explained before, this process executes subtle adjustments in the chosen harmony. In the proposed GHSC algorithm the PAR stands for the probability that the actual sample is assigned to the cluster of its roughly *nearest* sample is assigned to.
– RSR: as can be inferred from the above descriptions, none of the previous processes involves a variation in the number of clusters. The RSR will therefore modify this number and, in order to ensure that the variation in a harmony from one iteration to the next one is gradual, this parameter is applied to each harmony, not to each note. The RSR has two operation modes in which the possibility of increasing or decreasing the number of clusters is given with 50% of probability each; in the first case, a new cluster is created by splitting one of the actual solution (chosen randomly), and in the second case, one randomly selected cluster is eliminated and its samples are reassigned to the closest remaining cluster.

C. Local Search: this procedure is utilized to find local optima in the vicinity of a certain harmony. Specifically, this method measures the metric obtained when a determinate sample is assigned to all the other clusters in the solution and selects the best cluster for each sample. The process is applied to the harmonies under a certain probability and is sequentially repeated until all samples are assigned to their most optimal cluster.

D. Fitness evaluation: at every iteration the quality evaluation of newly improvised harmonies is made based on an objective function. To this end two widely known metrics will be utilized: the Davis-Bouldin (DB) index [18] and the Silhouette coefficient [19]. The Davis-Bouldin index is defined for a given cluster arrangement $\boldsymbol{\mathcal{X}}^*$ and a similarity metric $d(\cdot,\cdot)$ as

$$\mathrm{DB}(\boldsymbol{\mathcal{X}}^*) \doteq \frac{1}{K}\sum_{k=1}^{K}\max_{k\neq k'}\left\{\frac{\sum_{x\epsilon\boldsymbol{\mathcal{X}}_k^*} d^2(x,\mu_k)+\sum_{x\epsilon\boldsymbol{\mathcal{X}}_{k'}^*} d^2(x,\mu_{k'})}{d^2(\mu_k,\mu_{k'})}\right\}, \tag{2}$$

with μ_k denoting the centroid of cluster $\boldsymbol{\mathcal{X}}_k^*$. This index favors solutions with small distances between the samples assigned to the same cluster and large distances among different clusters. It is important to note that compact and well separated clusters entail low values of the DB index. Furthermore, this score does not present a monotonic behavior with the number of clusters, so it also allows validating the optimal number of clusters for a given data set. On the other hand, the Silhouette coefficient is a commonly utilized measure in clustering problems as it allows evaluating the quality not only of a single solution, but also that of each of the arranged clusters. For the n-th sample $\mathbf{X}_n$ assigned to cluster $k(n)$, the Silhouette coefficient $\varUpsilon_n$ is defined as

$$\varUpsilon_n \doteq \frac{\alpha_{k(n)}-\beta_{k(n)}}{\max\{\alpha_{k(n)},\beta_{k(n)}\}}, \tag{3}$$

where the parameter $\alpha_{k(n)}$ represents the average distance between samples in the cluster $\boldsymbol{\mathcal{X}}^*_{k(n)}$ (i.e. the intra-cluster distance) to which the n-th sample belongs, whereas $\beta_{k(n)}$ stands for the minimum distance between the samples in cluster $\boldsymbol{\mathcal{X}}^*_{k(n)}$ to the remaining samples assigned to different clusters $k' \neq k(n)$. For a cluster $\boldsymbol{\mathcal{X}}^*_k$, the silhouette coefficient is defined as the average of the Silhouette coefficients of its constituent samples. By using this coefficient good partitions featuring compact and well-defined clusters while separated from each other are obtained when its value gets close to its maximum value. The evaluation of these metric functions and their comparison to the fitness of harmonies from previous iterations permits to update the HM with the φ best harmonies.

E. Stop criterion: the search process stops when a fixed number of iterations $\mathtt{I}$ is reached. This criterion has been established in order to provide a fair comparison between the algorithms compared in the later discussed experiments.

Besides the novel encoding solution for avoiding redundancies presented in Section 2, two additional concepts are included in the proposed GHSC scheme. The first one is related with the differential characteristic of the defined operators, thus no population-based knowledge is used and a new harmony is improvised from its state at previous iteration. On the other hand, the improvisation operators are defined on the basis of the structural relationship of the feature space as provided by the Euclidean distances between samples and their closest neighbors. Last but not least, the fact that only feasible solutions are improvised during the iterative process guarantees that the computational complexity of the algorithm is reduced.

3 Experiments and Results

In this section different experiments based in synthetic scenarios obtained from two public repositories are presented. The proposed GHSC is compared to different clustering solutions presented in the literature:

1. The K-means algorithm [20], which can be regarded as one of the most popular clustering algorithms. This approach requires the number of clusters K as an input parameter, and it obtains a partition of the data into K clusters. It is fairly well known that the K-means is simple and easy to use, which motivates its widespread use in a large variety of problems. However, it is important to remark that the K-means obtains poor results in problems where clusters have different sizes, densities and/or many outliers. Additionally, this deterministic algorithm has a strong dependency with the initialization of the centroids which can make the algorithm get local optimum solutions. As last claim, the required input of the number of clusters to be discovered by this clustering method limits significantly its practicality. In these experiments, the well-known elbow method is used to automatically set the number of clusters K.

2. Density Based Spatial Clustering of Applications with Noise (DBSCAN [21]), which finds the number of clusters based on the assumption that the spatial density of samples belonging to a certain cluster should be higher than a predefined density-reachability threshold parameter (ϵ). DBSCAN requires an additional input parameter to set the minimum number of points N_{min} required to form a cluster. As points are assigned to clusters based on ϵ, N_{min} and the distance between samples $d(\cdot, \cdot)$, this algorithm does not need to know a priori the number of clusters to be sought. Additionally, DBSCAN is able to recognize clusters with different shapes and sizes, and takes into account the notion of noise, in such a way that outlier samples do not influence the algorithm's performance. However, it performs poorly in clustering problems over spaces with areas characterized by significant density differences, and still requires two input parameters (ε and N_{min}) to be configured.

3. Grouping Genetic Algorithm (GGA) [12], which shares the same encoding with the CHS with the exception that GGA also utilizes the s_y part. In connection to the general scheme of the GHSC detailed in Section 2.2 the initialization step (A), local search (C), metric evaluation (D) and stop criterium (E) are kept the same in the GGA approach. However, the GGA operators follow the selection, crossover and mutation mechanisms featured by evolutionary algorithms: first, a rank-based roulette wheel mechanism is adopted for the *selection* of the individuals to be mated. It is important to note that this rank-based selection mechanism is static, in the sense that probabilities of survival do not depend on the generation, but on the position of the individual in the list. Likewise, the *crossover* operator implemented in GGA is a modified version of the one initially proposed by Falkenauer [10], adapted to the clustering problem to remove empty clusters and fulfill the redundancy-minimizing encoding strategy explained in Section 2.1. Additionally, the GGA implements two different types of mutation: 1) cluster splitting, by which a selected cluster is split in two, and 2) clusters merging, which gathers two randomly selected groups into one.

3.1 Results and Discussion

This section discusses results obtained by the above algorithms in datasets with diverse characteristics in terms of density, size and/or shape, which eases the understanding and assessment of the weaknesses and strong points of all the algorithms within the benchmark. Besides these synthetic scenarios, experiments with the widely utilized `Iris` and `Wine` databases are also presented. All reported scores have been computed over 20 Monte Carlo experiments and `I` = 100 iterations. Forthcoming discussions are held on the *average* Rand Index (avg-RI) values, a measure of similarity between two clustering arrangements that in this case, is computed between the clustering produced by each algorithm and the *gold standard* known for every dataset. The proposed GHSC approach is configured with $\varphi = 50$, HMCR = 0.3, PAR = 0.1 and RSR = 0.2.

As anticipated above the discussion is focused on analyzing how GHSC performs in several feature spaces with different cluster properties, namely:

- `Spherical`, which is composed by $N = 300$ samples drawn uniformly at random from 8 independent Gaussian distributions with means μ_1=(-1, 0), μ_2 = (-1, -1), μ_3 = (-1, -3), μ_4 = (3, -1), μ_5 = (-1, 1), μ_6 =(2, -2), μ_7= (1, 2), μ_8= (3, 1), statistical independence between dimensions and standard deviation per dimension equal to 0.35.

– `Structured`, formed by $N = 400$ samples randomly generated using a Gaussian distribution from 3 classes with probability $p_1 = 0.5$, $p_2 = 0.33$ and $p_3 = 0.17$. Means for each class are μ_1= (0, 2), μ_2= (-1, -1) and μ_3= (2, -1), with the following covariance matrices:

$$\Sigma_1 = \begin{pmatrix} 1^2 & 0 \\ 0.8^2 & 0 \end{pmatrix}, \quad \Sigma_2 = \begin{pmatrix} 0.6^2 & 0 \\ 0.4^2 & 0 \end{pmatrix}, \quad \Sigma_3 = \begin{pmatrix} 0.3^2 & 0 \\ 0.5^2 & 0 \end{pmatrix}. \tag{4}$$

– `Unbalanced`, which comprises $N = 200$ samples randomly generated using a Gaussian distribution from 9 equiprobable classes, with means μ_1= (1, -1), μ_2= (-1.5, 0), μ_3= (0, 1), μ_4= (-1, 1), μ_5= (2, -1), μ_6= (-2, -1), μ_7= (-0.5, 2), μ_8= (-1, -1) and μ_9= (1.5, 0), statistical independence between dimensions and standard deviation per dimension equal to 0.2.

– `Iris`, which considers 3 classes formed by 50 samples each, totaling $N = 150$ instances. The challenge consists of differentiating among three different Iris plants (*Sentosa*, *Virginica* and *Versicolor*) based on 4 characteristics of the flowers: length and width of sepal and petal of the flower, all in centimeters.

– `Wine`, formed by 3 classes comprising 59, 41 and 78 samples each ($N = 178$ samples). These samples represent classes of wine from different regions of Italy that are defined by 13 analyzed chemical properties of the wines.

Table 1. Comparison of the results obtained by the proposed GHSC algorithm and GGA with DB and Υ index, DBSCAN and K-means algorithms in the considered clustering problems. Best scores are highlighted in bold.

Algorithm	`Spherical`		`Structured`		`Unbalanced`		`Iris`		`Wine`	
	K	avg-RI	K	avg-RI	K	avg-RI	K	avg-RI	K	avg-RI
K-means	9	0.949	4	0.875	3	0.780	3	0.880	3	0.702
DBSCAN	7	0.955	1	0.376	8	0.395	2	0.777	2	0.349
GGA (DB)	8	0.981	3	0.917	9	**1.000**	3	0.873	3	0.731
GGA (Υ)	7	0.957	3	0.951	9	0.993	3	0.899	3	0.722
GHSC (DB)	8	0.994	3	0.942	9	**1.000**	3	0.901	3	**0.739**
GHSC (Υ)	8	**0.995**	3	**0.965**	9	0.976	3	**0.903**	3	0.733

Table 1 summarizes the results obtained by the algorithms over the considered datasets. Results of the proposed GHSC and GGA are calculated by using the Davies-Bouldin (DB) index and the Silhouette (Υ) coefficient as their fitness function. It is important to recall that scores are given in terms of the Rand Index, computed by assuming that the best data partition is the one corresponding to the original classes (distributions) of the dataset at hand. Focusing on the `Spherical` dataset, the solution provided by GHSC improves on average the best results obtained with the GGA, DBSCAN and K-means by 1.4%, 4% and 4.5%, respectively. Solutions obtained with the DB index with GGA and GHSC are similar to each other (just 1.3% in favour of GHSC), whereas this gap widens an additional 4% when utilizing the Silhouette index Υ. This slight margin improvement must be assessed jointly with the fact that just GGA (DB) and both GHSC proposals manage to infer the correct number of clusters within the data. Finally, when comparing the best approaches (GGA and GHSC), the latter attains the best results regardless the index utilized (DB or Υ).

In the `structured` case the best result is obtained by GHSC with Υ index as its fitness function. It is interesting to notice the poor result obtained by the DBSCAN approach. In addition, neither K-means nor DBSCAN determine the actual number of clusters given by the statistical distributions used for generating this dataset. When evaluating the DB and Υ indexes, the latter scores better average results for both GGA and GHSC, from which it is concluded that this index adapts better to non-compact clusters.

When it comes to the `Unbalanced` dataset both GHSC and GGA with the DB index are able to split the cluster space according to the distributions from which instances were produced (avg-RI equal to 1.000). The CHS driven by the Υ coefficient as its fitness function also attains a perfect cluster arrangement. In this case, K-means underestimates the number of clusters and ultimately produces bad performance scores. From these experiments two conclusions can be drawn: 1) meta-heuristic approaches are a good alternative to optimally detect the number of clusters; and 2) the DB index outperforms Υ, i.e. the best approach when dealing with non-overlapping cluster spaces is to balance inter-cluster and intra-cluster distances (as done by the DB index) instead of focusing on the cluster dispersion targeted by the Υ index.

For the `Iris` dataset the best result is given by GHSC with Υ index. Nevertheless, both GGA and K-means obtain good solutions, slightly worse (1%-3%) than the GHSC with the DB index, but better than the solution provided by GGA with the DB index. The DBSCAN approach, though, does not provide an accurate solution for this experiment. This dataset is composed by one compact, well-defined cluster (suitable for the DB index) and two more disperse additional groups (appropriate for the Υ index). There lies the rationale why both indexes attain similar scores for the proposed GHSC method. Finally, a same line of reasoning can be taken for the `Wine` setup: the GHSC approach with the DB index scores best for this dataset outperforming GGA, K-means and DBSCAN. From this discussion an essential conclusion can be obtained: in general all meta-heuristic approaches are able to determine optimally the number of clusters and outperform traditional clustering schemes, with the proposed GHSC technique as the prevailing option in the benchmark.

4 Concluding Remarks

In light of the obtained results the most straightforward conclusions point out that the K-means algorithm is likely to fall in local optimums and is mostly useful for problems where clusters have compact spherical shapes of uniform sizes and densities. On the other hand, DBSCAN is not able to differentiate accurately overlapped clusters. However, both GGA [12] and the proposed GHSC have proven to optimally infer the number of clusters and the partitions, being the latter slightly better than the former. One of the improvements provided by GHSC when compared to other algorithms from the literature is the inference of the number of clusters. Unlike other algorithms that require a dedicated step to this purpose or further information regarding the dataset at han, the proposed algorithm is able to determine both the optimal partitioning of the samples and the number of clusters for each dataset embedded within the search process. The algorithm begins with a number of randomly chosen clusters, and converges towards the optimum number of groups using a measure of clustering quality as the objective function to optimize.

In this context, the measures utilized for the experimental part of this work – namely, the DB index and the Silhouette index – focus on establishing compact clusters with delimited shapes and no overlap. As such, the DB index measures the average similarity between each cluster and the one that most resembles it based on the intra/inter cluster distance ratio. The Silhouette index is based on the concepts of average scattering for clustering and the total separation among clusters, which also reflect the compactness of clusters. Apart from these commonalities between both indexes, the first one works better with overlapped clusters and in general is more stable for all kind of data sets. Apart from failing with overlapped clusters, the related literature (see e.g. [22,23,24] and references therein) has often concluded that the DB index attains lower values in general cluster spaces than the Silhouette index. Although the study presented in this paper for testing the performance of GHSC is a small representation of all existing techniques and types of datasets, the conclusions drawn in this reduced experimental setup are congruent with those by previous references and span their applicability over the meta-heuristic field. Future developments will gravitate on the application of GHSC to real clustering problems, the use of alternative coefficients as an objective function for the heuristic search and the implementation of this algorithm in massively parallel Big Data computing architectures.

Acknowledgments

This work has been supported in part by the Basque Government through the ELKARTEK program (ref. KK-2015/0000080).

References

1. Xu, R., Wunsch, D.: Survey of Clustering Algorithms. IEEE Transactions on Neural Networks 16(3), 645–678 (2005)
2. Johnson, S. C.: Hierarchical Clustering Schemes. Psychometrika 32(3), 241–254 (1967)
3. Davidson, I., Wagstaff, K. L., Basu, S.: Measuring Constraint-Set Utility for Partitional Clustering Algorithm. Springer Lecture Notes in Computer Science 4213, 115–126 (2006)
4. Al-Shultan, K. S.: A Tabu Search Approach to the Clustering Problem. Pattern Recognition 28(9), 1443–1451 (1995)

5. Du, K. L.: Clustering: A Neural Network Approach. Neural Networks 23, 89–107, (2010)
6. Dhillon, I. S., Guan, Y., Kulis, B.: Kernel K-means: Spectral Clustering and Normalized Cuts. ACM SIGKDD, 551–556 (2004)
7. Sarvari, H., Khairdoost, N., Fetanat, A.: Harmony Search Algorithm for Simultaneous Clustering and Feature Selection. International Conference of Soft Computing and Pattern Recognition, 202–207 (2010)
8. Hoang, D. C., Kumar, R.: A Robust Harmony Search Algorithm Based Clustering Protocol for Wireless Sensor Network. IEEE International Conference on Communications, 1–5 (2010)
9. Senthilnath, J., Kulkarni, S., Raghuram, D. R., Sudhindra, M., Omkar, S. N.: A Novel Harmony-Search Approach for Clustering Problems. Int. J. Swarm Intelligence 2(1) (2016)
10. Falkenauer, E.: A New Representation and Operators for Genetic Algorithms Applied to Grouping Problems. Evolutionary Computation, 123–144 (1994)
11. Geem, Z. W., Kim, J.-H., Loganathan, G. V.: A New Heuristic Optimization Algorithm: Harmony Search. Simulation 76(2), 60–68 (2001)
12. Agustín-Blas, L.E., Salcedo-Sanz, S., Jiménez-Fernández, S., Carro-Calvo, L., Del Ser J. and Portilla-Figueras, J.A.: A New Grouping Genetic Algorithm for Clustering Problems. Expert Systems with Applications 39(10), 9695–9703 (2012)
13. Falkenauer, E.: A Hybrid Grouping Genetic Algorithm for Bin Packing. Journal of Heuristics 2(1), 5–30 (1996)
14. Falkenauer, E.: A Genetic Algorithm for Bin Packing and Line Balancing. IEEE International Conference on Robotics and Automation 2, 1186–1192 (1992)
15. Landa-Torres, I. , Manjarrés, D., Salcedo-Sanz, S., Del Ser, J., Gil-López, S.: A Multiobjective Grouping Harmony Search Algorithm for the Optimal Distribution of 24-hour Medical Emergency Units. Expert Systems with Applications 40(6), 2343–2349 (2013)
16. Manjarres, D., Landa-Torres, I., Gil-Lopez, S., Del Ser, J., Bilbao, M. N., Salcedo-Sanz, S., Geem, Z. W.: A Survey on Applications of the Harmony Search Algorithm. Engineering Applications of Artificial Intelligence 26(8), 1818–1831 (2013)
17. Landa-Torres, I., Del Ser, J., Salcedo-Sanz, S., Gil-López, S., Portilla-Figueras, J. A., Alonso-Garrido, O.: A Comparative Study of Two Hybrid Grouping Evolutionary Techniques for the Capacitated p-Median Problem". Computers & Operations Research 39(9), 2214–2222 (2012)
18. Davies, D., Bouldin, D.: A Cluster Separation Measure. IEEE Transactions on Pattern Analysis and Machine Intelligence 1(2), 224–227 (1997)
19. Rousseeuw, P.: Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis. J. Comput. Appl. Math. 20, 53–65 (1987)
20. McQueen, J.: Some Methods for Classification and Analysis of Multivariate Observations. Berkeley Symposium on Mathematics and Statistics, 281–297 (1968)
21. Ester, M., Kriegel, H. P., Sander, J.: A Density-based Algorithm for Discovering Clusters in Large Spatial Data Bases with Noise. International Conference on Knowledge Discovery and Data Mining, 226–231 (1996)
22. Rendón, E., Abundez, I. M., Gutierrez, C., Zagal, S. D., Arizmendi, A., Quiroz, E. M., Arzate, H. E.: A Comparison of Internal and External Cluster Validation Indexes. American Conference on Applied Mathematics, 158–163 (2011)
23. Guerra, L., Robles, V., Bielza, C., Larraaga, P.: A Comparison of Cluster Quality Indices using Outliers and Noise. Intelligent Data Analysis 16, 703–715 (2012)
24. Bandyopadhyay, S., Saha, S.: Unsupervised Classification: Similarity Measures, Classical and Metaheuristic Approaches, and Applications. Springer (2013)