Dynamic Fuzzy Clustering using Harmony Search with Application to Image Segmentation

Osama Moh’d Alia\textsuperscript{1}, Rajeswari Mandava\textsuperscript{1}, Dhanesh Ramachandran\textsuperscript{1}, Mohd Ezane Aziz\textsuperscript{2}
\textsuperscript{1} CVRG - School of Computer Sciences, University Sains Malaysia, 11800 USM, Penang, Malaysia
\textsuperscript{2} Department of Radiology - Health Campus, Universiti Sains Malaysia, 16150 K.K, Kelantan, Malaysia
\{osama,mandava,dhaneshr@cs.usm.my\}, drezane@kb.usm.my

Abstract—In this paper, a new dynamic clustering approach based on the Harmony Search algorithm (HS) called DCHS is proposed. In this algorithm, the capability of standard HS is modified to automatically evolve the appropriate number of clusters as well as the locations of cluster centers. By incorporating the concept of variable length in each harmony memory vector, DCHS is able to encode variable numbers of candidate cluster centers at each iteration. The PBMF cluster validity index is used as an objective function to validate the clustering result obtained from each harmony memory vector. The proposed approach has been applied onto well known natural images and experimental results show that DCHS is able to find the appropriate number of clusters and locations of cluster centers. This approach has also been compared with other metaheuristic dynamic clustering techniques and has shown to be very promising.

Index Terms—dynamic fuzzy clustering, harmony search, image segmentation, PBMF cluster validity index

I. INTRODUCTION

Clustering is a typical unsupervised learning technique used to group similar data points according to some measurement of similarity. This measurement seeks to minimize the inter-cluster similarity while maximizing the intra-cluster similarity [1]. Clustering algorithms have been applied successfully in many fields such as machine learning, artificial intelligence, pattern recognition, web mining, image segmentation, biology, remote sensing, marketing, etc (see [2] and references therein). Due to this wide applicability, clustering algorithms have proven to be very reliable, especially in categorization tasks that calls for semi or full automation.

Clustering algorithms can be generally categorized into two groups; hierarchical, and partitional [1]. The former basically produces a nested series of partitions whereas the latter does clustering with one partitioning result. According to [3], the latter is more popular than the former in pattern recognition applications because the hierarchical clustering suffers from some drawbacks such as static behavior i.e., data points assigned to a cluster cannot move to another cluster, and the probability of failing to separate overlapping clusters.

Partitional clustering can further be subdivided into two categories: crisp or hard clustering where each data point belongs to only one cluster, and fuzzy or soft clustering which allows a data point to simultaneously belong to more than one cluster at the same time, based on some fuzzy membership grade. Fuzzy clustering is considered more appropriate than crisp clustering when the datasets such as images exhibit unclear boundaries between clusters or regions.

However, partitional clustering algorithms suffer from two major shortcomings. They are:

1) Prior knowledge of number of clusters (called cluster validity [4]): Dealing with real-world data, the number of clusters \(c\) is often unclear. Hence, appropriate \(c\) values have to be manually supplied a prior. This task is often very crucial and if not properly executed, might yield undesirable results;

2) Trapped in local optima: Many partitional techniques are sensitive to cluster centers initialization step. This sensitivity problem is inherited from greedy behavior of these algorithms. Therefore, a tendency to be trapped in local optima is very high [5], [6]. These issues will often cause undesirable clustering results.

These two problems have been the subject of several research efforts in last few decades as described in related work section. One way to overcome these shortcomings is to use metaheuristic optimization algorithms as clustering techniques. This approach is feasible and practical due to the NP-hard nature of partitional clustering problems [7]. Metaheuristic population-based algorithms are widely believed to be able to solve NP-hard problems with satisfactory near-optimal solutions and significantly less computational time compared to exact algorithms. Although many metaheuristic algorithms for solving fuzzy clustering problems have been proposed, the results are still unsatisfactory [8]. Therefore, an improvement to the optimization algorithms for solving clustering problems is still required.

In this paper, a new approach called Dynamic fuzzy Clustering using the Harmony Search (DCHS) is proposed. This approach takes advantage of the search capabilities of the metaheuristic Harmony Search (HS) algorithm [9] to overcome the local optima problem through its ability to explore the search space in an intelligent combination of local and global search. At the same time DCHS is able to automatically determine the appropriate number of clusters without any prior knowledge.

Several clustering algorithms have been successfully applied in the image segmentation problems [10], [11], [12], since the image segmentation can be modeled as a clustering problem [13]. Therefore, the effectiveness of our proposed algorithm, DCHS is evaluated on six natural grayscale images. Indeed, DCHS works as an image segmentation technique that
subdivides the test image into constituent regions. Moreover, a comparative study with the state of art algorithms in the domain of dynamic clustering is also considered.

The remainder of the paper is organized as follows: Section II describes previous related works. Section III reviews fuzzy partitional clustering. Section IV discusses the DCHS algorithm. Section V presents and discusses the experimental results. In the final section, conclusion and future directions to the proposed work are presented.

II. RELATED WORK

Partitional clustering algorithms assume that the number of clusters in unlabeled data is known and specified a priori. Selecting the suitable number of clusters is a non-trivial task and can lead to undesirable results if improperly executed. Therefore, many research works have been devoted to handle this matter. Commonly, three approaches were adopted and are described as follows:

1) The first approach applied the conventional quantitative evaluation functions, which is generally known as the cluster validity methods. This approach applies a given clustering algorithm to a range of \( c \) values, and evaluates the validity of the corresponding partitioning results in each case. Many criteria were developed to determine the cluster validity indices. These criteria have also spurred many different methods with different properties [14], [15], [16]. However, as mentioned by Dave & Krishnapuram [17], there are several problems with this approach, which can be summarized as:
   a) High computational demands, since the clustering algorithm must be performed for every value in the \( c \) range.
   b) The tendency to be trapped in the local optima and prone to initialization sensitivity.
   c) The cluster validity tests is affected if the tested data set has noise and outliers, hence, it could not be trusted.

2) These shortcomings motivate the researcher to investigate the second approach, where the clustering algorithm itself will determine the number of clusters. This approach evolved by modifying the objective function of the clustering algorithm i.e., fuzzy c-means algorithm or k-means and their variants algorithms. Some of these algorithms use split or merge methodologies to find the appropriate number of clusters. Such techniques initially start with a large or small number of clusters. Then, clusters are added or removed until they reach the most appropriate number of clusters such as [18], [19], [20], [21], [22], [23]. Even these algorithms are well known in clustering domain, but they are still sensitive to initialization and other parameters [24].

3) The third approach simultaneously solves the problems of automatic determination of number of clusters and initialization sensitivity, as well as getting trapped in local optima. This approach applies an optimization algorithm (such as Genetic algorithm, Particle Swarms algorithm, etc.) as a clustering algorithm (either hard or fuzzy) with the cluster validity index as its objective function. Such a combination leads to an approach that overcomes the aforementioned partitional clustering limitations in one framework. Examples of such combinations in the literature are: fuzzy variable string length genetic algorithm (FVGA) [25], Dynamic Clustering using Particle Swarm Algorithm (DCPSO) [8], Automatic Clustering Differential Evolution (ACDE) [26], Genetic Clustering for Unknown number of clusters K (GCUK) [27], Automatic Fuzzy Clustering Differential Evolution (AFDE) [28], Evolutionary Algorithm for Fuzzy Clustering (EAC-FCM) [29]. For further details and intensive survey, see [30] and for evolutionary algorithms for clustering problems, see [31].

From the literature, it is observed that the optimization approach is efficient in tackling the two aforementioned problems. Even that, an intensive work is needed to improve the results obtained from this approach. This paper, proposed an alternate optimization approach called DCHS to utilize the ability of HS algorithm as a clustering algorithm that can automatically determine the appropriate number of clusters of the test images and at a same time can avoid getting trapped in a local optima. HS algorithm works either locally, globally, or both by finding an appropriate balance between exploration and exploitation through its parameter settings [9], [32]. With this ability, the DCHS is able to avoid getting trapped in local optima and at the same time, overcome initialization sensitivity of cluster centers. Moreover, HS possess several advantages over traditional optimization techniques [32] such as:

1) HS is a simple population based metaheuristic algorithm and does not require initial value settings for decision variables;
2) HS uses stochastic random searches;
3) HS does not need derivation information;
4) HS has few parameters;
5) HS can be easily adopted in various types of optimization problems [33].

These features increase the flexibility of the HS algorithm in producing better solutions.

III. FUNDAMENTALS OF FUZZY CLUSTERING

In this section, a brief description of fuzzy clustering is provided. Where the clustering algorithm of a fuzzy partitioning type is performed on a set of \( n \) pixels \( X = \{x_1, x_2, \ldots, x_n\} \), each of which, \( x_i \in \mathbb{R}^d \), is a feature vector consisting of \( d \) real-valued measurements describing the features of the object represented by \( x_i \). Fuzzy clusters \( c \) of the pixels can be represented by a fuzzy membership matrix called fuzzy partition \( U = [u_{ij}]_{(c \times n)}, U \in M_{fcn} \) as in Eq (1). Where \( u_{ij} \) represents the fuzzy membership of the \( ith \) object to the \( jth \) fuzzy cluster. In this case, every data object belongs to a particular (possibly null) degree of every fuzzy cluster.
Harmony search (HS) is a new metaheuristic optimization method that imitates the music improvisation process where the musicians improvise their instruments’ pitch by searching for a perfect state of harmony [9]. It has been successfully tailored to different optimization problems (see [33] and references therein). Consequently, the HS algorithm provides a possibility of success in NP-hard problems such as clustering problems.

The DCHS algorithm is a model of HS algorithm that addresses the issue of automatically determining the appropriate number of clusters as well as identifying the location of the cluster centers in a given data set. The following sections provide a detailed description of the proposed algorithm.

A. Initialization of Harmony Memory (HM)

Each harmony memory vector encode the cluster centers of the test data set. However, since these cluster centers are unknown for the test data set, a possible range of number of clusters that the test data set may possess is tested. Consequently, each harmony memory vector can vary in length according to the randomly generated number of clusters for each vector. To initialize the HM with feasible solutions, each harmony memory vector initially encodes a number of cluster centers, denoted by \( ClustNo \), such that:

\[
clustNo = (rand() \times (clustMaxNo - clustMinNo)) + clustMinNo
\]  

where \( rand() \) is a function that generates a random number in \([0, 1]\), and \( clustMaxNo \) is an estimate of the maximum number of clusters (upper bound), while \( clustMinNo \) is the minimum number of clusters (lower bound). The values of upper and lower bounds are set depending on the data sets used. Therefore, the number of clusters \( ClustNo \) will range from \( clustMinNo \) to \( clustMaxNo \).

Even though the vector length is allowed to vary, for a matrix representation, each vector length in HM must be made equal to the maximum number of clusters \( clustMaxNo \). As a result, the remnants of unused vector elements (referred to as "don’t care" as in [34]) are represented with ‘#’ sign. For example, consider the case where the maximum number of clusters in the test data set is equal to 7, and one of the HM vector has only 3-candidate cluster centers. Then these 3 centers will be set in the vector in arbitrary order while the rest of the vector’s elements are set to don’t care with the ‘#’ sign as illustrated in Eq. (3).

It is worth mentioning here, that the fitness function for each harmony vector is calculated and saved in harmony memory as explained in section (IV-C).

\[
M_{fcn} = \left\{ U \in \mathbb{R}^{n \times m} | \sum_{j=1}^{m} U_{ij} = 1, 0 < \sum_{j=1}^{m} U_{ij} < n \right\}
\]

IV. THE DCHS ALGORITHM

The new harmony vector, \( a^{NEW} = (a_1^{NEW}, a_2^{NEW}, a_3^{NEW}, \ldots, a_n^{NEW}) \), is a vector with the candidate cluster centers, and the values of this vector is generated depending on the HS’s improvisation rules. This new harmony vector inherits the values of its components \( a^{NEW} \) from the harmony memory vectors stored in HM with the probability of Harmony Memory Consideration Rate (HMCR), otherwise, the value of the components of the new harmony vector is selected from the possible range with a probability of \((1-HMCR)\). Furthermore, the new vector components which are selected out of memory consideration operator are examined to be pitch adjusted with the probability of Pitch Adjustment Rate (PAR).

The other important issue worth mentioning is when the inherited components of the new harmony vector have don’t care values (‘#’). In this case, no pitch adjustment will take place. Fig. 1 shows the pseudo code of the improvisation step.

B. Improvise a New Harmony

Once the new harmony vector is generated, a count is done on the generated number of cluster centers in the new vector. If it is less than the minimum number of cluster centers \( (clustMinNo) \), the new vector will be rejected. Otherwise, the new vector will be accepted and a fitness function is computed using a cluster validity measurement described in Section (IV-C). Then, the new vector is compared with the worst harmony memory solution in terms of the fitness function. If it is better, the new vector is included in the harmony memory and the worst harmony is excluded. This process is repeated until the maximum number of iterations (NI) is reached. In the end, the best solution among the maximum value of fitness function of each HM solution vectors is selected to be the best solution vector.

C. Evaluation of Solutions

The evaluation (fitness value) of each harmony memory vector indicates the degree of goodness of the solution it represents. In order to evaluate the goodness of each harmony memory vector, the empty components that may appear in the harmony vector is removed and the remaining components which represent the cluster centers are used to cluster (or segment) the test image. Each pixel in the image data set will be assigned to one or more clusters with a membership grade. The fuzzy membership value for each pixel is calculated as in [4] as follows:

\[
u_{ij} = \frac{1}{\sum_{c=1}^{c} \left( \frac{\|x_i - v_k\|}{\|x_i - v_k\|} \right)^{2/\gamma}}
\]

where \( \{v_k\}_{k=1}^{c} \) are the centroids of the clusters \( c \) and \( \| \| \) denotes an inner-product norm (e.g. Euclidean distance) from
\begin{verbatim}
while (i ≤ clustMaxNo) do
  if (rand ∈ (0, 1) ≤ HMCR) then
    choose a value from HM for i
  if (i ≠ \#i) then
    end if
  if (rand ∈ (0, 1) ≤ PAR) then
    adjust the value of i by:
    \( i_{new} = i_{old} + rand \in (0, 1) \times (0.0001 \times maxVisVal) \)
  end if
end while

Fig. 1. The improvisation step of DCHS Algorithm
\end{verbatim}

the data point \( x_i \) to the \( j \)th cluster center, and the parameter \( m \in [1, \infty) \), is a weighting exponent on each fuzzy membership that determines the amount of fuzziness of the resulting classification.

After that, the goodness of the clustering result is measured using a cluster validity index. Therefore, the validity index measurement is used as the fitness function in this study.

Several indices for the fuzzy clustering assessment have been proposed (e.g. see [14], [15], [16] and references therein). In this paper, a recently developed index, which exhibits a good trade-off between efficacy and computational concern, is used. This index is named PBMF-index which is the fuzzy version of PBM-index [35]. PBMF-index is defined as follows:

\[
PBMF(c) = \left( \frac{1}{c^p} \times \frac{E_c}{E_c} \times D_c \right)^2
\]

where \( c \) is the number of clusters. Here

\[
E_c = \sum_{j=1}^{c} \sum_{i=1}^{n} u_{ij}^m \| x_i - v_j \|
\]

and

\[
D_c = max_{i,j} \| v_i - v_j \|
\]

where \( v_i \) is the center of the \( i \)th cluster, the power \( p \) is used to control the contrast between the different cluster configurations and it is set to be 2. E1 is a constant term for a particular data set and it is used to avoid the index value from approaching zero. The value of \( m \), which is the fuzziness weighting exponent, is experimentally set to 1. \( D_c \) measures the maximum separation between two clusters over all possible pairs of clusters, while \( E_c \) measures the sum of \( c \) within-cluster distances (i.e., compactness).

The maximum value of PBMF-index indicates correct clustering results and therefore the correct number of clusters that could be gained. Consequently, maximization of the fitness function is desirable to reach the near-optimal solution. It is also worth mentioning here that according to Pakhira et al. [35] the maximum value for \( c \) cluster centers allowed is \( \sqrt{n} \) which is considered as a safe measurement to avoid monotonic behavior of this index.

For the purpose of simplification and time complexity reduction, a simple and compact representation of a data set is implemented. The simplification process is based on finding the frequency of each pixel in the tested image, therefore the image is represented as \( X = (x_1, h_1), \ldots, (x_i, h_i), \ldots, (x_q, h_q) \) where \( h_i \) is the frequency of occurrence \( x_i \) in the image, and \( q \) is the total number of distinct \( x \) value in the image, where \( \sum_{i=1}^{q} h_i = n \). As a consequence, the dimensions of the partition matrix are reduced.

To illustrate this idea, assume a gray image with 8 bit resolution and size of \( 512 \times 512 \), then typically there are only 256 possible values for the pixel. Therefore, the value of \( n \) becomes 256 instead of 262144 and the partitioning matrix becomes \( U = c \times 256 \) instead of \( U = c \times 262144 \).

Therefore, PBMF-index could be rewritten as follows:

\[
PBMF(c) = \left( \frac{1}{c} \times \frac{E_1}{E_c} \times D_c \right)^2
\]

V. EXPERIMENTAL RESULTS

Experiments were conducted using several natural images: clouds, peppers, science magazine and robot. In addition, one MR brain image and one satellite image of Mumbai city (India) were used. All of these grayscale images (shown in Fig.2) have been used to show the wide applicability of the proposed approach and to compare its results with results obtained from well-known algorithms in this domain: AFDE, FVGA, ACDE, DCPSE, GCUK, and classical DE.

In order to achieve best results from any optimization algorithm, the appropriate selection of algorithm’s parameter values is required; since these parameters will seriously affect the performance and accuracy of the algorithm. Therefore, the selection of harmony search related parameters (i.e. Harmony Memory Size (HMS), HMCR, PAR, Number of Improvisation NI) is a very important step. In this paper, these values are experimentally set as shown in Table I. Besides, the value of maximum number of clusters (upper bound) is set to 10, while the minimum number of clusters (lower bound) is set to 2.

All the experiments were performed on Intel core 2 due 2.66GHz processor with 2GB of RAM; while the codes were written using Matlab 2008a.

The optimal number of clusters for any image is set to a range instead of a specific value, since the determination of number of clusters in any image (except artificial images) is subjective [36]. For that, the optimal range of the tested images are set based on two factors: first, based on the visual analysis survey conducted by a group of people as in [8], [36], secondly, the observations on the optimal number of clusters for each image and it variations in different works, such as Das and Konar [28], where they set the optimal number of
clusters for pepper image to 4 while Das et al. [26] set the optimal number of clusters for pepper image to 7.

Table II shows the results of DCHS against AFDE [28] and FVGA [25] algorithms. The numbers for AFDE and FVGA are obtained from [28], whose experimental setup we have repeated for the sake of comparison. From the results, it can be seen that AFDE failed to obtain the correct results in data set 5 (Mumbai). FVGA on the other hand performed badly on data sets 1 and 6 (Clouds and Robots). Both AFDE and FVGA were also out of the cluster range in data set 4 (Science Magazine). However, it appears that DCHS always finds a solution within the optimal range. These results clearly show the efficiency of DCHS, and that it works better than the other algorithms based on the optimal range set in this paper.

Table III shows results from another experimental setup. The details of the experiment are as explained in [26]. The data set used is similar to that of Table II, with the exclusion of the MR images. DCHS is compared with other algorithms such as ACDE [26], DCPSO [8], GCUK [27] and Classical DE [28]. From this table, it can be seen that DCHS works better than some of these algorithms according to the optimal range of clusters. DCHS always finds a solution within the optimal range while some algorithms such as ACDE (on the Mumbai data set), DCPSO (on the Robot data set) and GCUK (on the Peppers, Science Magazine and Mumbai data sets) fail to obtain results within the cluster ranges.

Finally, the execution time of DCHS algorithm to find the near optimal number of cluster centers using typical image representation is around 1 hour and 34 minutes depending on the image size and depth, while the image with a simplified representation takes around 5 seconds only. This is a significant improvement in the execution time. This simplified representation makes the proposed dynamic clustering algorithm more efficient.

VI. Conclusions

In this paper, the problem of finding a globally optimal partition of a set of images into unknown number of clusters is studied. A novel algorithm, named DCHS is proposed, where clustering is modeled as an optimization problem. In the proposed algorithm, a new harmony memory representation incorporating the concept of variable-length harmony vectors of image clusters is employed. These have enabled DCHS to specify the appropriate number of image clusters without any prior knowledge. The experimental results on six different images have shown that DCHS produces higher quality solutions in comparison to other state-of-the-art algorithms in the dynamic clustering domain. It can be concluded that DCHS algorithm outperforms most algorithms for these images. For future work, the hybridization of DCHS algorithm with heuristic components will be explored in order to improve the performance of the existing algorithm.

VII. Acknowledgment

This research is supported by ‘Universiti Sains Malaysia Research University Grant’ grant titled ‘Delineation and visualization of Tumour and Risk Structures - DVTRS’ under grant number 1001 / PKOMP / 817001.

References

TABLE II
MEAN AND STANDARD DEVIATION OF 30 RUNS OF THE AUTOMATIC CLUSTERING RESULTS OBTAINED BY DCHS OVER SIX GRAYSCALE IMAGES.

<table>
<thead>
<tr>
<th>Image</th>
<th>Optimal clusters range</th>
<th>DCHS</th>
<th>ACDE</th>
<th>DCPSO</th>
<th>GCUK</th>
<th>Classical DE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clouds</td>
<td>3-4</td>
<td>3.425±0.594</td>
<td>3.15±0.211</td>
<td>4.50±0.132</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Peppers</td>
<td>4-8</td>
<td>5.67±0.730</td>
<td>5.05±0.428</td>
<td>5.25±0.212</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Science magazine</td>
<td>2-4</td>
<td>5.15±0.660</td>
<td>4.15±0.772</td>
<td>5.25±0.982</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mumbai</td>
<td>3-6</td>
<td>4.65±1.292</td>
<td>6.20±0.479</td>
<td>4.85±2.489</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Robot</td>
<td>3-4</td>
<td>3.57±1.152</td>
<td>3.05±0.076</td>
<td>2.25±0.908</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

TABLE III
MEAN AND STANDARD DEVIATION OF 40 RUNS OF THE AUTOMATIC CLUSTERING RESULTS OBTAINED BY DCHS OVER FIVE GRAYSCALE IMAGES.

<table>
<thead>
<tr>
<th>Image</th>
<th>Optimal clusters range</th>
<th>DCHS</th>
<th>ACDE</th>
<th>DCPSO</th>
<th>GCUK</th>
<th>Classical DE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clouds</td>
<td>3-4</td>
<td>3.425±0.594</td>
<td>4.15±0.211</td>
<td>4.50±0.132</td>
<td>3.00±0.000</td>
<td></td>
</tr>
<tr>
<td>Peppers</td>
<td>4-8</td>
<td>5.15±0.660</td>
<td>7.05±0.038</td>
<td>6.85±0.064</td>
<td>3.90±0.447</td>
<td>8.50±0.067</td>
</tr>
<tr>
<td>Science magazine</td>
<td>2-4</td>
<td>2.575±0.813</td>
<td>4.05±0.772</td>
<td>3.25±0.082</td>
<td>6.35±0.093</td>
<td>3.50±0.059</td>
</tr>
<tr>
<td>Mumbai</td>
<td>3-6</td>
<td>4.65±1.292</td>
<td>6.10±0.079</td>
<td>4.65±0.674</td>
<td>7.45±0.043</td>
<td>5.25±0.007</td>
</tr>
<tr>
<td>Robot</td>
<td>3-4</td>
<td>3.57±1.152</td>
<td>4.25±0.428</td>
<td>2.30±0.012</td>
<td>3.35±0.982</td>
<td>3.00±0.004</td>
</tr>
</tbody>
</table>


