Quantum Behaved Particle Swarm Optimization for Data Clustering with Multiple Objectives

Heyam Al-Baity

Computer Science Department
University of Birmingham, UK &
King Saud University, Riyadh, KSA
hha090@cs.bham.ac.uk

Souham Meshoul

Computer Science Department
University Constantine 2
smeshoul@umc.edu.dz

Ata Kaban

Computer Science Department
University of Birmingham, UK
A.Kaban@cs.bham.ac.uk

Lilac AlSafadi

IT Department
King Saud University, Riyadh, KSA
lalsafadi@ksu.edu.sa

Abstract—Clustering is an important tool in many fields such as exploratory data mining and pattern recognition. It consists in organizing a large data set into groups of objects that are more similar to each other than to those in other groups. Despite its use for over three decades, it is still subject to a lot of controversy. In this paper, we cast clustering as a Pareto based multi-objective optimization problem which is handled using a quantum behaved particle swarm optimization algorithm. The search process is carried out over the space of cluster centroids with the aim to find partitions that optimize two objectives simultaneously, namely compactness and connectivity. Global best leader selection is performed using a hybrid method based on sigma values and crowding distance. The proposed algorithm has been tested using synthetic and real data sets and compared to the state of the art methods. The results obtained are very competitive and display good performance both in terms of the cluster validity measure and in terms of the ability to find trade-off partitions especially in the case of close clusters.

Keywords- multi objective optimization, quantum behaved particle swarm optimization, clustering, F-measure.

I. INTRODUCTION

Recent advances in information technology have fostered the creation of high quantities of data. These data sets are more often unstructured and therefore difficult to analyze. In order to gain some insight about the content of the data, to summarize data into useful information, to find out the natural grouping of a set of patterns or to discover hidden patterns, data clustering has been widely used. Data clustering, also known as clustering analysis, is the process of partitioning a dataset into meaningful groups (clusters), such that the objects in the same group are similar to each other and dissimilar to the objects in the other groups. It is considered as an unsupervised classification problem where classes are not known in advance [1][2]. Clustering is one of the most important techniques in data mining and has been applied to many interesting applications such as machine learning [3], bioinformatics [4], document classification and web mining [5]. The main objective of clustering is to segment a large set of data into meaningful clusters so that the intra-cluster homogeneity and the inter-cluster heterogeneity are both maximized. A cluster can be viewed as a set of points that is compact and isolated [6]. There are many clustering methods that have been proposed in the literature over the past decades. Comprehensive reviews

may be found in [6, 7]. K-means is one of the most popular and widely used algorithms for data clustering. As most of the conventional clustering algorithms, it attempts to optimize just a single clustering criterion. However, such criterion may not be able to capture the intended notion of clusters given the diverse characteristics of the datasets [7]. Additionally, the quality of clustering resulting from partitioning algorithms depends highly on the initial settings of the centroids which may lead to locally optimal solutions. A common solution to the latter problem is to perform multiple runs of the algorithm with different initial centroids and then select the best partitioning results as the final clustering solution. However, this approach is not effective when dealing with a large dataset and a large number of clusters [8].

In order to overcome the above problems and to obtain a good and meaningful clustering, global search optimization techniques such as Genetic algorithms (GAs) and Particle Swarm Optimization (PSO) can be employed to explore the search space and achieve better quality solutions. Moreover, the clustering solutions should be assessed from different aspects or different validity criteria, rather than a single aspect. Therefore, tackling the clustering problem as a truly multi-objective optimization problem would be a promising attempt in order to improve the quality of the final clustering solutions. To some extent, it has been investigated in multiobjective context using evolutionary algorithms [9, 10] but not that much using swarm based multiobjective algorithms.

In [11, 12], we proposed the general framework for multi-objective optimization that extends the single objective Quantum behaved Particle Swarm Optimization (QPSO) algorithm proposed by Sun et al. [13] to unconstrained and constrained problems with multiple objectives respectively. QPSO is a swarm based optimization algorithm designed for solving continuous problems. Unlike Particle Swarm Optimization, in QPSO, the particles move in the search space according to quantum mechanics principles. QPSO uses only one tunable parameter and it is easy to implement. The developed multiobjective QPSO (MOQPSO) is characterized by its global leader selection strategy used to update particles' positions. It is a Pareto based algorithm that has shown very competitive results on function optimization.

In this paper, we investigate solving the data clustering problem through MOQPSO in order to find several possible

partitions of a data set according to multiple objectives, namely compactness and connectivity. The objective of this work is twofold. We demonstrate on one hand the ability of MOQPSO to handle the clustering problem, and on the other hand the ability of our multi-objective clustering to obtain meaningful tradeoff partitions.

The remainder of the paper is organized as follows: In Section II, the key features of the proposed multi-objective clustering algorithm are described. Section III describes the data sets used for experiments and the evaluation of cluster validity. It also reports on experiments conducted and the results obtained. Finally, conclusions and perspectives are provided in section IV.

II. KEY FEATURES OF THE PROPOSED MOQPSO FOR CLUSTERING

Given a data set X consisting of n points, $X = \{x_1, x_2, \dots, x_n\}$, multiobjective clustering is the task that consists in finding the set of non-dominated partitions or clusters $\vec{C}_j^* = (c_{j1}^*, c_{j2}^*, \dots, c_{jk}^*)$ that optimizes (minimizes or maximizes) a vector of objective functions which measures the quality of a partition from different aspects. Each c_i denotes a cluster and k is the number of clusters [8, 9]. Each partition satisfies the following constraints:

1. $\forall_i c_i \neq \emptyset$
2. $\forall_{i,j} c_i \cap c_j = \emptyset \quad \text{for } i \neq j$
3. $\bigcup_{i=1}^k c_i = X$

A. Clustering Encoding

In order to solve clustering with multiobjective QPSO, a suitable way of particles encoding should be adopted in order to represent the potential clustering solutions. As we propose an approach that is based on the principle of partitional clustering, each particle position is related to the centroids of the clusters as shown in figure 1. In other words, the potential solution in the proposed MOQPSO clustering algorithm represents a partition or clustering which is given by a set of clusters centroids. Therefore, the search is performed in the space of centroids.

Figure 1 (a) A dataset of 5 points with three attributes. (b) The proposed encoding of particle positions, when the number of clusters is $k=2$.

B. The objective functions

The performance of a multi-objective clustering algorithm depends heavily on the selection of the clustering objectives

[8]. In this work, compactness and connectivity have been chosen as the two complementary objectives to be optimized as they can measure the clustering quality from different aspects. Additionally, these two objectives can balance each other so that trivial clustering solutions could be avoided and interesting regions in the search space could be explored [9].

Cluster Compactness

This validity measure computes the overall deviation of a clustering by finding the overall sum of the distances between data points and their cluster centers. It is calculated according to the following equation given in [9]:

$$Comp(C) = \sum_{C_k \in C} \sum_{i \in C_k} \delta(i, \mu_k)$$

where C is the set of clusters or partitions, μ_k is the centroid of cluster C_k , and $\delta(\cdot, \cdot)$ is the selected distance function (Euclidean distance in our case). This objective should be minimized as it tends to keep the intra-cluster variation small [9].

Cluster Connectedness

This measure is based on the idea that the neighboring data points should be placed in the same cluster. It is computed by the following equation [9]:

$$Conn(C) = \sum_{i=1}^N \left(\sum_{j=1}^L x_{i,nn_{ij}} \right)$$

$$\text{where } x_{r,s} = \begin{cases} \frac{1}{j} & , \text{if } \nexists C_k : r \in C_k \wedge s \in C_k \\ 0 & , \text{otherwise} \end{cases}$$

where nn_{ij} denotes the j^{th} nearest neighbor of point i , N is the size of the dataset, and L is the number of neighbors of point i . This objective should also be minimized.

C. Outline of MOQPSO for Clustering

Let X be an input dataset composed of points to be grouped into k clusters. Each point in X is defined by q attributes. Each attribute represents a decision variable. Therefore the dimension of the problem is $D=k*q$. Solving the clustering problem by the MOQPSO algorithm using the particle's position encoding described above requires first an initialization step where initial partitions are derived from k randomly generated centroids. Then points in the dataset X are assigned to the closest centroids in terms of Euclidean distance.

For this purpose, a procedure called *Assign_Points* is designed. This procedure allows mapping between the space of centroids and the space of points to cluster correspondences. It allows us to derive a partition from knowing the cluster centroids and the dataset X . It is described by the algorithm given below.

The compactness and the connectivity values of each obtained partition are computed and then an initial set of non-dominated solutions is created and set as the archive of the global best solutions or the current Pareto clustering solutions. Let us denote the set of non-dominated solutions or the archive of the global best solutions in Pareto sense encountered at iteration t by A^t . S^t refers to the swarm of particles at iteration t .

Assign_Points

Input: $clusters\_centroids$
Dataset X

```

For  $i=1$  to  $|X|$ 
  Set  $distance\_to\_cluster$  to Empty;
  For  $j=1$  to number_of_clusters
     $distance\_to\_cluster(j)=Distance(X(i), clusters\_centroids(j))$ ;
  EndFor
   $partition(i) = \text{cluster number}(\min(distance\_to\_cluster))$ ;
EndFor
Output: partition
END

```

As outlined by the algorithm shown below, MOQPSO undergoes an iterative process after the initialization step. During this iterative process, the positions of the particles are updated according to QPSO dynamics. Each iteration starts with computing the mean-best solution (step 1) which is the mean over all self-best positions of particles. Then, each particle's position is updated according to equation 2 or 3 shown below. This update operation starts by computing the particle's attractor (step 4) according to the following equation that shows the update of the j^{th} dimension of the i^{th} position:

$$p_{ij}^t = \phi_{ij}^t * sbest_{ij}^t + (1 - \phi_{ij}^t) * gbest_{ij}^t \quad (1)$$

where ϕ_{ij}^t is random value generated from a uniform distribution over the range $[0,1]$, $sbest$ refers to the particle self-best position and $gbest$ to the global best leader of the particle. This latter is selected from the archive using the hybrid method we developed in [11]. According to this selection strategy, the nearest neighbor solutions to the current particle are first identified based on their sigma values [14]. Then the less crowded one, using the crowding distance method [15], is chosen as the global leader. As can be seen in equations (2) and (3), the update of the position is influenced by the local attractor of the particle and the mean-best position. β is the contraction expansion parameter. It is the only one tunable parameter in QPSO. Once a new position vector is recorded, a new partition is derived using the above Assign_Points procedure and evaluated in terms of compactness and connectivity. At step 11, the self-best position of a particle is updated based on the Pareto dominance relation. The current position is compared with the self-best position and the non-dominated one is kept as the current $sbest$. If both are non-dominated, one of them is selected randomly as the current $sbest$. Once

all positions have been updated, the new archive at step 12 is determined by keeping the non-dominated solutions among those belonging to the current archive A^t and the current swarm S^t . Parameter β is decreased linearly as shown in equation 4 at step 13 and the process undergoes a new iteration until a termination criterion is satisfied. At the end of this process, the obtained Pareto optimal set along with the corresponding Pareto front are given as the output of the algorithm. The proposed MOQPSO for data clustering can be described as follows:

Input: X, K, q

- $D = k * q$ // Problem dimension
- N = population size
- S^0 = Initialize positions of particles (clusters centroids) randomly
- point_In_Cluster = Assign_Points (ClustersCentroids, X)
- $sbest_i$ = Initialize self best position of particle P_i for $i=1..N$
- T = maximum number of iterations
- F_i = evaluate Particle P_i for $i=1..N$
- A^0 = Initial set of non-dominated partitions
- $\beta = \beta_{max}$
- $t = 0$

Repeat

1. Compute mean best position
2. **For** each particle P_i
3. $gbest = \text{Select-leader}(A^t, P_i)$;
4. $p_i^t = \text{Compute local attractor}$
5. $u_i^t = \text{rand}(0,1)$;
6. **For** $j=1$ to D
7. **if** ($\text{rand}(0,1) > 0.5$) **Then**
8. $x_{ij}^{t+1} = p_{ij}^t + \beta \cdot |mbest_j^t - x_{ij}^t| \cdot \ln(1/u_{ij}^t)$ (2)
9. **Else**
9. $x_{ij}^{t+1} = p_{ij}^t - \beta \cdot |mbest_j^t - x_{ij}^t| \cdot \ln(1/u_{ij}^t)$ (3)
- Endif**
- EndFor**
10. Evaluate (Assign_Points(x_i, X))
11. Update self best position of particle P_i
- End for**
12. $A^{t+1} = \text{update\_Archive}(A^t, S^t)$
13. $\beta^t = (\beta_{max} - \beta_{min}) * \frac{T-t}{T} + \beta_{min}$ (4)
14. **Until** a maximum number of iterations T
15. **Output** A^T

III. EXPERIMENTS

The proposed algorithm has been implemented using MATLAB on a modern PC (CPU Intel Core i7 3.2 GHZ, RAM 6 GB, GPU: Intel HD Graphics, Windows 8 professional 64 bits). It has been applied to both hand crafted and real world data sets. The hand crafted data sets are generated by a Gaussian cluster-generator described in [16]. Real data sets are taken from the UCI machine learning databases repository [17]. Each data set is defined by a tuple of parameters ($k, q, Size$) where k denotes the number of clusters, q is the dimension of dataset point, $Size$

is the dataset size [18]. Synthetic data consist of the following sets with their corresponding parameters: 2d4c(4,2,1123), 2d10c(10,2,520) and 10d10c (10,10,436). The real data sets are: Ruspini (4,2,75), Iris (3,4,150) and Wisconsin (2,9, 699). In our study, an external validation measure called the F-measure [19] is used to assess the performance of the algorithm. The F-measure is one of the commonly used validity measures in the specialized literature. It measures the degree of similarity of an obtained clustering to each ground truth class of the given data set. Assume that $GC = (GC_1, GC_2, \dots, GC_k)$ denotes the ground truth classes of the dataset and $C = (C_1, C_2, \dots, C_k)$ denotes the obtained clustering result. Then the F-measure of cluster C_i and class GC_i is given by the following equation:

$$F(C_i, GC_j) = 2|C_i \cap GC_j| / |C_i| + |GC_j|$$

the overall F-Measure of a clustering is given by:

$$F(C, GC) = \sum_j \frac{|GC_j|}{|X|} \max_i F(C_i, GC_j)$$

The F-measure values are within the range [0,1]. The larger the F-measure values, the higher the quality of clustering.

Preliminary experiments have been conducted to find suitable settings of the algorithm's parameter, namely the contraction expansion parameter β which decreases linearly from 1.2 to 0.5. The number of iterations and number of particles were set to 200 and 150 respectively even good results have been obtained with smaller values in the case of 2d4c and Ruspini data sets.

Then, other experiments have been performed to assess the performance of the proposed algorithm from data clustering and multiobjective optimization perspectives. Qualitatively,

MOQPSO achieved very good partitions when compared to ground truth ones especially in the case of 2d4c, 10d10c and Ruspini. To conduct a quantitative assessment, the algorithm has been run 30 times for each dataset and the best solutions in terms of F-measure have been recorded. K-means algorithm has been run the same number of times on the same data sets. Then, statistical results have been collected using the obtained partitions of both algorithms and represented in terms of boxplots as shown in Figure 2. As can be seen from the boxplots, MOQPSO outperforms K-means in all data sets in terms of best values except for 2d10c dataset where K-means performed slightly better. Additionally, MOQPSO succeeded in achieving best median values in all data sets except for Iris data set. Moreover, our algorithm exhibits more stability than K-means. Another comparative study with other algorithms from the literature has been performed. Table 1 shows the results in terms of median and interquartile of F-measure using the proposed MOQPSO algorithm and four other algorithms, namely FCM (Fuzzy C-Means), PCA (Possibilistic Clustering Algorithm), IQEAC (Improved Quantum Evolutionary Algorithm for data Clustering) and GA (Genetic Algorithm) with settings as reported in [20, 21]. The best results have been obtained by our algorithm on 2d4c, 10d10c and Ruspini data sets. It also achieved very competitive results for the remaining data sets.

Furthermore, the obtained results reveal that the proposed algorithm is effective and flexible in providing a set of diverse partitioning solutions. For example, in the case of the 2d4c data set, the proposed algorithm provides several tradeoff clustering solutions. Figure 3 shows some of these Pareto clustering solutions. In all Pareto solutions, MOQPSO was able to find the core partitions. The difference between these partitions lies in the boundaries of the clusters. This is particularly interesting in the case of overlapping clusters.

Data sets	MOQPSO	FCM	PCA	IQEAC	GA
2d4c	0.9982(0.00)	0.9392(0.00)	0.7428(0.1877)	0.9784(0.00)	0.9730(0.00)
2d10c	0.9031(0.00)	0.8861(0.0681)	0.7320(0.1039)	0.9582(0.0066)	0.9027(0.0506)
10d10c	1.00(0.00)	0.9254(0.00)	0.6666(0.1122)	1.00(0.00)	0.9344(0.0353)
Iris	0.8396(0.0153)	0.8923(0.00)	0.7233(0.2367)	0.8988(0.00)	0.8923(0.0005)
Wisconsin	0.9241(0.0002)	0.9558(0.00)	0.6812(0.3685)	0.9677(0.00)	0.9662(0.0014)
Ruspini	1.00(0.00)	-	-	-	-

Table 1. Comparison other algorithms based on Median (Interquartile)

Figure 2 F-measure results using MOQPSO and K-means for all data sets

Figure 3 Pareto solutions representing the trade-off partitions for 2d4c data set

IV. CONCLUSIONS AND FUTURE WORK

In this paper, we have presented a multi-objective quantum behaved particle swarm optimization algorithm for solving partitional clustering problems. To our knowledge, it is the first Pareto-based MOQPSO specifically designed for this purpose. The main feature of this approach is its ability to provide the end users with multiple optimal clustering options from which a partition can be chosen according to their specific needs. Our algorithm has been found to perform successfully on both synthetic and real data sets. The proposed MOQPSO-based approach outperformed k-means in most cases and shows competitive results compared to other algorithms. These preliminary results are very promising and could be improved by considering the impact of other objective functions. As a scope of further research, a comparative study with other MOP algorithms such as MOPSO should be performed. Furthermore, the approach should be tested on more complex data sets that exhibit higher volume and dimension of data points as well as complex distributions with non-linearly separable clusters. Finally, a possible improvement to the current approach should be studied to include automatic detection of clusters.

REFERENCES

- [1] G. Gan, C. Ma, and J. Wu, "Data clustering theory, algorithms, and applications", Society for Industrial and Applied Mathematics, 2007.
- [2] L. Kaufman and P.J. Rousseeuw, "Finding groups in data: an introduction to cluster analysis", volume 344. Wiley, 2009.
- [3] C. Carpineto and G. Romano, "A lattice conceptual clustering system and its application to browsing retrieval". *Machine Learning*, 24(2), pp. 95–122, 1996.
- [4] P. Bertone and M. Gerstein, "Integrative data mining: the new direction in bioinformatics". *Engineering in Medicine and Biology Magazine, IEEE*, 20(4), pp. 33–40, 2001.
- [5] G. Mecca, S. Raunich, and A. Pappalardo, "A new algorithm for clustering search results". *Data & Knowledge Engineering*, 62(3), pp. 504–522, 2007.
- [6] A.K. Jain, "Data clustering: 50 years beyond k-means". *Pattern Recognition Letters*, 31(8), pp. 651–666, 2010.
- [7] E. Hruschka, R.J.G.B. Campello, A.A. Freitas, and A.C.P.L.F. de Carvalho, "A survey of evolutionary algorithms for clustering". *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 39(2), pp. 133–155, 2009.
- [8] U. Maulik, S. Bandyopadhyay, and A. Mukhopadhyay, "Multiobjective Genetic Algorithms for Clustering: Applications in Data Mining and Bioinformatics". Springer, 2011.
- [9] J. Handl and J. Knowles, "An evolutionary approach to multiobjective clustering". *IEEE Transactions on Evolutionary Computation*, 11(1), pp. 56–76, 2007.
- [10] S. Bandyopadhyay, U. Maulik, and A. Mukhopadhyay, "Multiobjective genetic clustering for pixel classification in remote sensing imagery". *IEEE Transactions on Geoscience and Remote Sensing*, 45(5), pp. 1506–1511, 2007.
- [11] H. AlBaity, S. Mesoul, and A. Kaban, "On extending quantum behaved particle swarm optimization to multi-objective context". In *Proceedings of the IEEE World Congress On Computational Intelligence (IEEE CEC 2010)*, pp. 996–1003, 2012.
- [12] H. Al-Baity, S. Meshoul, A. Kaban, "Constrained Multi-Objective Optimization using a Quantum Behaved Particle Swarm". In *Neural Information Processing (ICONIP 2012)*, Part III, LNCS7665, Springer-Verlag Berlin Heidelberg, (2012) 456–464.
- [13] J. Sun, B. Feng, and W. Xu, "Particle swarm optimization with particles having quantum behavior". In *Proceedings of the IEEE Congress on Evolutionary Computation*, 2004.
- [14] S. Mostaghim and J. Teich, "Strategies for finding good local guides in multi-objective particle swarm optimization (mopso)". In *Proceedings of the IEEE Swarm Intelligence Symposium*, pp. 26–33, 2003.
- [15] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGAII". *IEEE Transactions on Evolutionary Computation*, 6(2), pp. 182–197, 2002.
- [16] J. Handl and J. Knowles, "Improvements to the scalability of multi-objective clustering". In *Proceedings of The 2005 IEEE Congress on Evolutionary Computation (IEEE CEC 2005)*, Vol. 3, pp. 2372–2379, 2005.
- [17] C. Blake and C.J. Merz. *UCI repository of machine learning databases*. 1998.
- [18] C. Ramdane, S. Meshoul, M. Batouche, and M. Kholadi, "A quantum evolutionary algorithm for data clustering". *International Journal of Data Mining, Modeling and Management*, Vol. 2(4), pp. 369–387, 2010.
- [19] E. Rendón, I. Abundez, A. Arizmendi, and E.M. Quiroz, "Internal versus external cluster validation indexes". *International Journal of Computers and Communications*, 5(1), pp. 27–34, 2011.
- [20] C. Ramdane and M.K. Kholadi, "Improvements to the quantum evolutionary clustering". *Int. J. Data Analysis Techniques and Strategies*, Vol. 5(2), pp. 175–197, 2013.