Evolutionary and Iterative Crisp and Rough Clustering I: Theory

Manish Joshi[1] and Pawan Lingras[2]

[1] Department of Computer Science, North Maharashtra University
Jalgaon, Maharashtra, India
joshmanish@gmail.com

[2] Department of Mathematics and Computing Science, Saint Mary's University
Halifax, Nova Scotia, B3H 3C3, Canada
pawan@cs.smu.ca

Abstract. Researchers have proposed several Genetic Algorithms (GA) based clustering algorithms for crisp and rough clustering. In this two part series of papers, we compare the effect of GA optimization on resulting cluster quality of K-means, GA K-means, rough K-means, GA rough K-means and GA rough K-medoid algorithms. In this first part, we present the theoretical foundation of the transformation of the crisp clustering K-means and K-medoid algorithms into rough and evolutionary clustering algorithms. The second part of the paper will present experiments with a real world data set, and a standard data set.

Keywords: Crisp Clustering, Rough Clustering, GA Rough K-means, GA Rough K-medoid.

1 Introduction

Clustering is an unsupervised learning process that partitions physical or abstract objects into groups based on some optimality criterion (e.g. similarity). The typical partition based clustering algorithms, K-means and K-medoids, categorize an object into precisely one cluster. Whereas, fuzzy clustering [1][13] and rough set clustering [5][12][14][16] provide ability to specify the membership of an object to multiple clusters, which can be useful in real world applications.

Deterministic local search converges to the nearest local optimum from a starting position of the search, hence K-means result largely depends on the initial cluster centers. On the contrary, stochastic search heuristics inspired by evolution and genetics have the ability to cope with local optima by maintaining, recombining and comparing several candidate solutions simultaneously. Use of GAs for clustering is proposed by [2][9][11]. GA guided evolutionary algorithms are also proposed for rough set and fuzzy clustering [8][12][15].

In this paper we discuss various aspects of crisp, rough set based, and evolutionary clustering algorithms. We discuss and present appropriate modifications required to the basic K-means and K-medoid algorithms so that these algorithms adapt to rough and evolutionary clustering. In particular, we explain K-means,

S. Chaudhury et al. (Eds.): PReMI 2009, LNCS 5909, pp. 615–620, 2009.

GA K-means, rough K-means, GA rough K-means, and GA rough K-medoid algorithms and their corresponding fitness functions.

Section 2 describes how rough set theory is embedded with a K-means algorithm. Section 3 discusses the inclusion of GA to K-means, rough K-means, and rough K-medoid algorithms. We also discuss the formulation of a fitness function for GA based crisp and rough clustering in this section, followed by the conclusion in section 4. Due to space restrictions, an elaborate experimental comparison is presented separately in the second part of the series.

2 Rough Clustering

In addition to clearly identifiable groups of objects, it is possible that a data set may consist of several objects that lie on the fringes. The conventional clustering techniques mandate that such objects belong to precisely one cluster. Such a requirement is found to be too restrictive in many data mining applications [4]. In practice, an object may display characteristics of different clusters. In such cases, an object should belong to more than one cluster, and as a result, cluster boundaries necessarily overlap. Fuzzy set representation of clusters, using algorithms such as fuzzy C-means, makes it possible for an object to belong to multiple clusters with a degree of membership between 0 and 1 [13]. In some cases, the fuzzy degree of membership may be too descriptive for interpreting clustering results. Rough set based clustering provides a solution that is less restrictive than conventional clustering and less descriptive than fuzzy clustering.

Lingras and West [5] provided an alternative based on an extension of the K-means algorithm [3][10]. Incorporating rough sets into K-means clustering requires the addition of the concept of lower and upper bounds. The rough K-means approach has been a subject of further research. Peters [14] discussed various refinements of Lingras and West's original proposal [5]. These included calculation of rough centroids and the use of ratios of distances as opposed to differences between distances similar to those used in the rough set based Kohonen algorithm described in [6]. The rough K-means and its various extensions [12], [14] have been found to be effective in distance based clustering. However, there is no theoretical work that proves that rough K-means explicitly finds an optimal clustering scheme. Moreover, the quality of clustering that is maximized by the rough clustering is not precisely defined. We compare crisp and rough clustering algorithm results and present our observations in the second part of this paper.

Rough K-means Algorithm. We represents each cluster $c_i, 1 \leq i \leq k$, using its lower $\underline{A}(c_i)$ and upper $\overline{A}(c_i)$ bounds. All objects that are clustered using the algorithm follow basic properties of rough set theory such as:

(P1) An object $\boldsymbol{x}$ can be part of at most one lower bound

(P2) $\boldsymbol{x} \in \underline{A}(\boldsymbol{c}_i) \Longrightarrow \boldsymbol{x} \in \overline{A}(\boldsymbol{c}_i)$

(P3) An object $\boldsymbol{x}$ is not part of any lower bound

$$\Updownarrow$$

$\boldsymbol{x}$ belongs to two or more upper bounds.

```
Input:
    $k$: the number of clusters,
    $D(n, m)$: a data set containing n objects where each object has m dimensions,
    $p$: a threshold value (1.4),
    $w\_lower$: relative importance assigned to lower bound (0.75),
    $w\_upper$: relative importance assigned to upper bound (0.25),
Output:
    A set of clusters.Each cluster is represented by the objects in the lower region
    and in boundary region (upper bound)
Steps:
    arbitrarily choose $k$ objects from $D$ as the initial cluster centers (centroids);
    repeat
        (re)assign each object to lower/upper bounds of appropriate clusters by
            determining its distance from each cluster centroid,
        update the cluster centroids using number of objects assigned and relative
            importance assigned to the lower bound and upper bound of the cluster;
    until no change;
```

Fig. 1. The K-means algorithm for rough clustering

Fig. 1 depicts the general idea of the algorithm. An object is assigned to lower and/or upper bound of one or more clusters. For each object vector, $\boldsymbol{v}$, let $d(\boldsymbol{v}, \boldsymbol{c}_j)$ be the distance between itself and the centroid of a cluster $\boldsymbol{c}_j$. Let $d(\boldsymbol{v}, \boldsymbol{c}_i) = \min_{1 \le j \le k} d(\boldsymbol{v}, \boldsymbol{c}_j)$. The ratios $d(\boldsymbol{v}, \boldsymbol{c}_i)/d(\boldsymbol{v}, \boldsymbol{c}_j)$, $1 \le i, j \le k$, are compared with a cut-off value to determine the membership of an object $\boldsymbol{v}$. This parameter is called as a *threshold*. Let $T = \{j : d(\boldsymbol{v}, \boldsymbol{c}_i)/d(\boldsymbol{v}, \boldsymbol{c}_j) \le threshold \text{ and } i \neq j\}$.

1. If $T \neq \emptyset$, $\boldsymbol{v} \in \overline{A}(\boldsymbol{c}_i)$ and $\boldsymbol{v} \in \overline{A}(\boldsymbol{c}_j), \forall j \in T$. Furthermore, $\boldsymbol{v}$ is not part of any lower bound. The above criterion guarantees that property (P3) is satisfied.
2. Otherwise, if $T = \emptyset$, $\boldsymbol{v} \in \underline{A}(\boldsymbol{c}_i)$. In addition, by property (P2), $\boldsymbol{v} \in \overline{A}(\boldsymbol{c}_i)$.

It should be emphasized that the approximation space A is not defined based on any predefined relation on the set of objects. The lower and upper bounds are constructed based on the criteria described above.

The values of p(a threshold), *w_lower*, *w_upper* are finalized based on the experiments described in [7].

3 Evolutionary Clustering Algorithms

This section contains some of the basic concepts of genetic algorithms as described in [2]. A genetic algorithm is a search process that follows the principles of evolution through natural selection. The domain knowledge is represented using a candidate solution called a *chromosome*. Typically, a chromosome is a single *genome* represented as a vector of length n:

$$g = (g_i \mid 1 \le i \le n), \tag{1}$$

where g_i is called a *gene*.

A group of chromosomes is called a *population*. Successive populations are called *generations*. A generational GA starts from initial generation $G(0)$, and each generation $G(t)$ generates a new generation $G(t+1)$ using genetic operators such as *mutation* and *crossover*. The mutation operator creates new genomes by changing values of one or more genes at random. The crossover operator joins segments of two or more genomes to generate a new genome.

The evaluation process of a genome i.e. evaluate $G(t)$, is a combination of two steps. The first step determines membership of all objects to corresponding clusters. As described in earlier section, appropriate calculations for crisp and rough clustering figure out the members of each cluster. In the next step, fitness of the genome is determined. The intuitive distance measure is used to decide the fitness of the genome. Obviously, there is a difference between the fitness calculations for crisp clustering and rough clustering. The fitness formulas used for both clustering are described below.

Genome Fitness Function for Crisp Clustering. The fitness is calculated based on the allocation of all objects to the clusters. It is given by:

$$Fitness \ = \ \sum_{i=1}^{k} \sum_{u \in c_i} d(u, x_i). \tag{2}$$

$Fitness$ is the sum of the Euclidean distances for all objects in the cluster; u is the point in space representing a given object; and x_i is the centroid/medoid of cluster c_i (both u and x_i are multidimensional). The function d computes distance between any two vectors u and v using following equation.

$$d(u, v) \ = \ \sqrt{\sum_{j=1}^{m} (u_j \ - \ v_j)^2}. \tag{3}$$

Here, the value of m indicates the total number of dimensions.

Genome Fitness Function for Rough Clustering. The Fitness function has to change to adapt to the rough set theory by creating lower and upper versions of the Fitness as:

$$\underline{Fitness} \ = \ \sum_{i=1}^{k} \sum_{u \in \underline{A}(c_i)} d(u, x_i), \tag{4}$$

$$\overline{Fitness} \ = \ \sum_{i=1}^{k} \sum_{u \in \overline{A}(c_i)} d(u, x_i), \tag{5}$$

where $\underline{A}(c_i)$ and $\overline{A}(c_i)$ represents lower and upper bound of cluster c_i. The distance function d does not change. The $Fitness$ value for the rough clustering is calculated as

$$Fitness \ = \ w\_lower \times \underline{Fitness} + w\_upper \times \overline{Fitness}. \tag{6}$$

where *w_lower* and *w_upper* are relative importances assigned to lower and upper bound of the clusters.

Thus evolutionary algorithms for clustering differ mostly in the genome representation and the fitness calculation. The variations of GA based algorithms for crisp and rough clustering are described in next subsections.

3.1 GA K-means

The crisp K-means algorithm is modified to adapt the principles of GA. The chromosome has a total of $k \times m$ genes. A batch of every m genes represents centroid of a corresponding cluster. The population size and generation values are input parameters.

3.2 GA Rough K-means

The proposed genome for the evolutionary algorithm has a total of $k \times m$ genes, where k is the desired number of clusters and m is the number of dimensions used to represent objects and centroids. The first m genes represent the first centroid. Genes $m + 1, \ldots, 2 \times m$ give us the second centroid, and so on. Finally, $((k-1) \times m) + 1, \ldots, k \times m$ corresponds to the k^{th} centroid. The rough fitness measure given by Eq. 6 is minimized.

3.3 GA Rough K-medoid

Unlike K-means algorithm where mean value is used as a centroid of a cluster, in K-medoid algorithm actual object is used as a reference point of a cluster [15]. A medoid is the most centrally located object in a given cluster. For k clusters, we have k medoids. A genetic algorithm is used to search for the most appropriate k medoids. The genome has k genes, each corresponding to a medoid. This reduces the size of a genome from $k \times m$ by a factor of m to k. The steps in this algorithm are similar to that of GA rough K-means. The major difference is the use of medoids instead of centroids of the clusters.

The gene values for rough K-medoids are discrete values corresponding to object IDs as opposed to continuous real variables used for centroids in the rough K-means evolution. This results in a restricted search space for the proposed rough K-medoids leading to further increase in the chances of convergence. We have tested this hypothesis in our second part of the paper.

4 Conclusion

In this paper - the first part of a two part series - we discuss how crisp clustering K-means and K-medoid algorithms adapt to rough and evolutionary clustering. The paper describes algorithmic steps for rough K-means, GA K-means, GA rough K-means and GA rough K-medoid. We also discuss the fitness function for rough and evolutionary algorithms to determine the quality of clustering. The second part of this paper will elaborate on an experimental comparison with a real world data set, and a standard data set.

References

1. Bezdek, J.C., Hathaway, R.J.: Optimization of Fuzzy Clustering Criteria using Genetic Algorithms (1994)
2. Buckles, B.P., Petry, F.E.: Genetic Algorithms. IEEE Computer Press, Los Alamitos (1994)
3. Hartigan, J.A., Wong, M.A.: Algorithm AS136: A K-Means Clustering Algorithm. Applied Statistics 28, 100–108 (1979)
4. Joshi, A., Krishnapuram, R.: Robust Fuzzy Clustering Methods to Support Web Mining. In: Proc. ACM SIGMOD Workshop Data Mining and Knowledge Discovery, June 1998, pp. 1–8 (1998)
5. Lingras, P., West, C.: Interval Set Clustering of Web Users with Rough K-Means. Journal of Intelligent Information Systems 23, 5–16 (2004)
6. Lingras, P., Hogo, M., Snorek, M.: Interval Set Clustering of Web Users using Modified Kohonen Self-Organizing Maps based on the Properties of Rough Sets. Web Intelligence and Agent Systems: An International Journal 2(3), 217–230 (2004)
7. Lingras, P.: Precision of rough set clustering. In: Chan, C.-C., Grzymala-Busse, J.W., Ziarko, W.P. (eds.) RSCTC 2008. LNCS (LNAI), vol. 5306, pp. 369–378. Springer, Heidelberg (2008)
8. Lingras, P.: Evolutionary rough K-means Algorithm. In: Proc. The Fourth International conference on Rough Set and Knowledge Technology, RSKT2009 (2009)
9. Lu, Y., Lu, S., Fotouhi, F., et al.: FGKA: A Fast Genetic K-Means Clustering Algorithm. In: Proc. ACM Symposium on Applied Computing (2004)
10. MacQueen, J.: Some Methods for Classification and Analysis of Multivariate Observations. In: Proceedings of Fifth Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 281–297 (1967)
11. Maulik, U., Bandyopadhyay, S.: Genetic Algorithm-Based Clustering Technique. Pattern Recognition 33, 1455–1465 (2000)
12. Mitra, S.: An Evolutionary Rough Partitive Clustering. Pattern Recognition Letters 25, 1449 (2004)
13. Pedrycz, W., Waletzky, J.: Fuzzy Clustering with Partial Supervision. IEEE Trans. on Systems, Man and Cybernetics 27(5), 787–795 (1997)
14. Peters, G.: Some Refinements of Rough k-Means. Pattern Recognition 39, 1481–1491 (2006)
15. Peters, G., Lampart, M., Weber, R.: Evolutionary rough k-medoid clustering. In: Peters, J.F., Skowron, A. (eds.) Transactions on Rough Sets VIII. LNCS, vol. 5084, pp. 289–306. Springer, Heidelberg (2008)
16. Peters, J.F., Skowron, A., Suraj, Z., et al.: Clustering: A Rough Set Approach to Constructing Information Granules, pp. 57–61 (2002)