Evolutionary and Iterative Crisp and Rough Clustering II: Experiments

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. In this second part of the paper, we compare the cluster quality of K-means, GA K-means, rough K-means, GA rough K-means and GA rough K-medoid algorithms. We experimented with a real world data set, and a standard data set using total within cluster variation, precision and execution time as the measures of comparison.

Keywords: Rough Clustering, Crisp Clustering, GA based Clustering, Cluster Quality.

1 Introduction

In this second part of our two part series, we discuss and compare results of crisp, rough set based, iterative and evolutionary clustering algorithms [3]. In particular, we compare performance of K-means, GA K-means, rough K-means, GA rough K-means, and GA rough K-medoid algorithms using appropriate evaluation metrics.

Latiff et al. [2] compared GA based algorithm with K-means and PSO in wireless sensor networks domain. Małyszkoa et al. [5] compared K-means and GA K-means algorithms for image segmentations. These two studies, like most of the other comparative studies, concentrated on crisp clustering. This paper compares evolutionary and iterative algorithms for both crisp as well as rough clustering using K-means and K-medoid clustering strategies.

Section 2 describes the evaluation metrics used for comparison. Comparative results for two different types of data set are presented and discussed in two subsequent sections, followed by the conclusions in section 5.

2 Evaluation Metrics

In order to compare the results obtained using evolutionary and iterative versions of crisp and rough clustering we use a *total within cluster variation* measure. The measure accumulates distances between a cluster center (cenroid/medoid) and objects assigned to a cluster.

S. Chaudhury et al. (Eds.): PReMI 2009, LNCS 5909, pp. 621–627, 2009.

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

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 provides the distance between two vectors. The distance $d(u,v)$ is given by:

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

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

The *Fitness* function has to adapt to the rough set theory by creating lower and upper approximations of the *Fitness* as:

$$\underline{Fitness} = \sum_{i=1}^{k} \sum_{u \in \underline{A}(c_i)} d(u, x_i), \quad (3)$$

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

where $\underline{A}(c_i)$ and $\overline{A}(c_i)$ represent lower and upper bounds of cluster c_i. The *Fitness* value for the rough clustering is calculated as

$$Fitness = w\_lower \times \underline{Fitness} + w\_upper \times \overline{Fitness}. \quad (5)$$

where *w_lower* and *w_upper* are relative importance assigned to lower and upper bound of the clusters. Smaller the value of fitness, better the cluster quality.

3 Comparative Results – Real World Data Set

We used the data obtained from a public library to compare the results of various algorithms. The data consist of books borrowed by members. The objective is to group members with similar reading habits. Information about how many times a member borrows books of a particular category is collected. The data is normalized in the range of 0 to 1 to reduce the effect of outliers. In order to visualize the data set, it is restricted to two dimensions. Data of 1895 members shows propensity of a member to borrow a book of a certain category. It will be interesting to see how all the algorithms carve out reasonable clusters from such scattered data.

The Gas used the crossover probability of 70% and mutation probability of 10%. We carried experiments with different population sizes and generations. Table 1

shows the results obtained using K-means and rough K-means algorithms. Clusters generated by rough K-means have less intra-cluster variation than for the clusters generated by K-means algorithm. Hence we can conclude that the rough K-means results in better cluster quality than the crisp K-means algorithm. We have observed similar trends for the synthetic data set.

Table 1 also shows the comparison of K-means against GA K-means and rough K-means against GA rough K-means algorithms. The results include average *Fitness* from five trials. For a normal range of population size and generations the GA K-means does not outperform the K-means. But for population size of 500 and 500 generations the average *Fitness* of GA K-means for 3 clusters is less than K-means. This performance improvement requires 90 seconds of computation. For five clusters the GA K-means (population size of 500 and 500 generations) could outperform K-means at the cost of 111.2 seconds of processing time.

As the data set size increases the population size and generations of GA should be increased to obtain improved *Fitness*. In some cases, the higher computational cost of GA K-means may not be justified by slight increase in accuracy.

Table 1 shows that GA rough K-means improves the results of rough K-means. Moreover, GA rough K-means does this with far less population size and generations than GA K-means.

Both GA rough K-medoid and GA rough K-means results are better than the K-means results. The GA rough K-medoid algorithm is faster than the GA rough K-means in surpassing the K-means results (see Table 2). GA rough K-means requires

Table 1. Comparison between crisp and evolutionary algorithms for Library data

Average K-means Fitness	Population size, Generations	GA K-means		Average Rough K-means Fitness	GA Rough K-means	
		Average Fitness	Avg. Time (Sec.)		Average Fitness	Avg. Time (Sec.)
3 Clusters						
7.9764	10, 20	9.1223	1	6.9517	8.094	1.2
	20, 20	8.6642	1		7.82	2.4
	20, 30	8.3182	1.4		7.12	2.4
	30, 50	8.0323	1.6		**6.91**	**4.4**
	50, 50	8.0261	2.8		6.9001	6
	100, 100	7.9780	6.8		6.9006	19
	500, 500	**7.9761**	**90**		-	-
5 Clusters						
5.9020	10, 20	8.9841	1	5.2061	7.1033	1.4
	20, 20	6.7930	1.8		6.7505	2.8
	20, 30	7.4541	1.8		6.2648	3
	30, 50	6.3575	2.8		5.8794	4.6
	50, 50	6.4889	3.2		5.2801	8
	100, 100	5.9527	16.4		**5.0730**	**25.8**
	500, 500	**5.9002**	**111.2**		**-**	**-**

Table 2. Comparison of K-means, GA rough K-means and GA Rough K-medoid for Library Data set

K-means Fitness	Population size, Generations	GA Rough K-means		GA Rough K-medoid	
		Fitness (Best / Average)	Avg. Time (Sec.)	Fitness (Best / Average)	Avg. Time (Sec.)
3 Clusters					
7.9764	10, 20	8.094	1.2	**7.7042**	**1.2**
	20, 20	7.82	2.4	7.1390	2.2
	20, 30	7.12	2.4	7.2670	2.8
	30, 50	**6.91**	**4.4**	7.0719	4
	50, 50	6.9001	6	7.0154	6.4
	100, 100	6.9006	19	7.0465	19
5 Clusters					
5.9020	10, 20	7.1033	1.4	6.0141	2
	20, 20	6.7505	2.8	**5.3114**	**3**
	20, 30	6.2648	3	5.3502	3.2
	30, 50	**5.8794**	**4.6**	5.1660	5
	50, 50	5.2801	8	5.1770	8.6
	100, 100	5.0730	25.8	5.1034	21.4

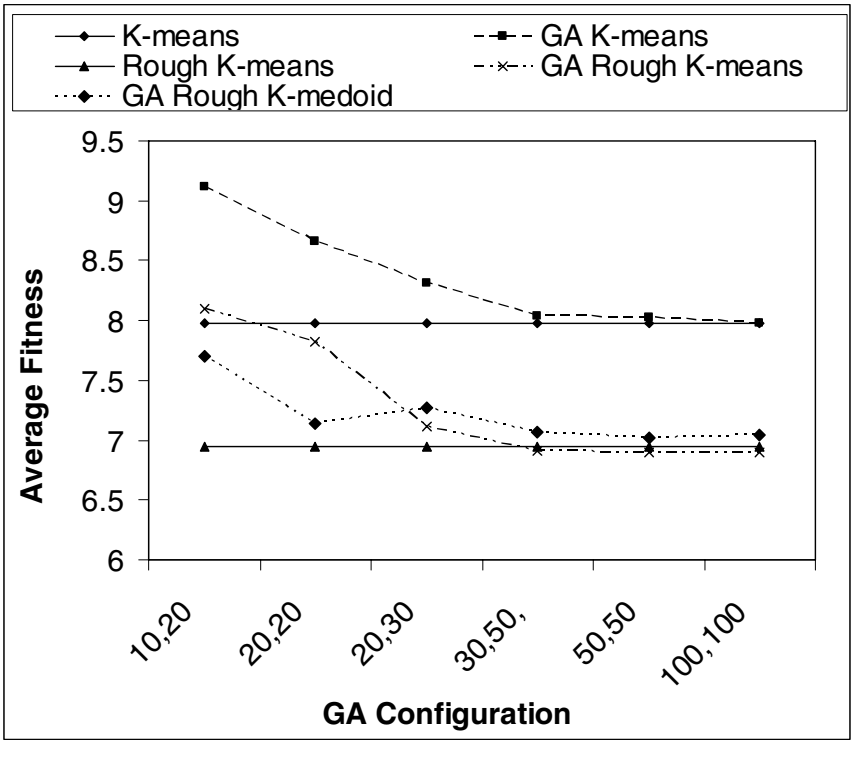

Fig. 1. Comparison chart of average fitness obtained by different algorithms

4.6 seconds with 30 population size and 50 generations to get better solution than K-means. Whereas GA rough K-medoid with 20 population size and 20 generations surpasses the K-means results in 3 seconds.

Fig. 1 shows the performance of GAs for different configurations. GA K-means performs better than K-means for large number of population size and generations. GA rough K-medoid algorithm promptly outdoes the K-means results, but for higher population size and generations GA rough K-means generates optimal results.

4 Comparative Results – Standard Data Set

We used *Letter Recognition* [1] data from the University of California Irvine machine learning repository. The data set contains 20,000 events representing character images of 26 capital letters in the English alphabet based on 20 different fonts. Each event is represented using 16 primitive numerical attributes (statistical moments and edge counts) that are scaled to fit into a range of integer values from 0 through 15.

In order to crosscheck the results and compare the cluster quality we decided to consider limited number of characters. We prepared a data set that consists of 8 distinctly different characters A, H, L, M, O, P, S, and Z. This data set consists of 6114 events.

Besides using the *Fitness* measure for comparison we calculated the average precision for each cluster. Each cluster is labelled using the character that appears most frequently in the cluster. Precision of the cluster is defined as the number of events that match the cluster label divided by the total number of events in the cluster. Average precision is the average of all cluster precisions.

Table 3 and Table 4 show the comparison of results obtained using K-means and rough K-means algorithms for an experimental data set.

Rough K-means clustering generates better quality clusters than the crisp K-means algorithm. This conclusion is supported by reduced *Fitness* and increase in average precision. Moreover, only one letter 'H' is not prominently identified by rough K-means whereas two letters 'H' and 'S' are not prominently identified by any clusters generated using K-means algorithm.

Table 3. K-means algorithm clustering for an experimental set of characters

Cluster No	Cluster Label Character	Frequency	Precision	Average Precision	*Fitness*
0	M	412/1119	0.37	0.60	406.23
1	Z	276/880	0.31		
2	Z	378/734	0.51		
3	P	607/639	0.95		
4	A	387/660	0.59		
5	A	311/514	0.61		
6	L	328/328	1.0		
7	O	600/1240	0.48		

Table 4. Rough K-means algorithm clustering for an experimental set of characters

Cluster No	Cluster Label Character	Frequency	Precision	Average Precision	*Fitness*
0	L	335/335	1.0	0.82	372.23
1	O	553/684	0.81		
2	Z	536/705	0.76		
3	A	183/255	0.72		
4	A	358/362	0.99		
5	S	96/294	0.33		
6	M	292/306	0.95		
7	P	553/555	1.0		

GA based algorithms however generate poor results for this standard data set. For crisp as well as for rough clustering, most of the objects are grouped into few clusters. The remaining clusters are left empty. Further investigations are necessary to determine the reasons for failure of GAs for the character data set.

5 Conclusions

In this second part of a two part series, we provide experimental comparison of results obtained by K-means, GA K-means, rough K-means, GA rough K-means and GA rough K-medoid algorithms. We applied all algorithms to a synthetic data set, a real world data set, and a standard data set. A simple and intuitive measure of total within cluster variation (Fitness) is used for the evaluation.

The rough K-means algorithm seems to provide better cluster quality in terms of the Fitness and average precision than the crisp K-means algorithm.

For sufficiently high population size and generations, GA K-means can improve average performance of K-means. As the size of data set increases, higher population size and generations are required in GA K-means algorithms to outperform the K-means results. Execution time increases when GAs with higher population size and generations are used. There is a trade off between execution time and better cluster quality when the GA K-means algorithm is used.

GA rough K-medoid converges faster and surpasses the K-means results with smaller populations and fewer generations than GA rough K-means. But for larger population and more generations the GA rough K-means results are superior to all other algorithms.

GA effect on rough clustering is more promising than that on crisp clustering. For both small as well as large data sets, the GA rough K-means and the GA rough K-medoid generate better clustering in reasonable amount of execution time.

Unexpectedly though, the GA version could not cope up with the data set that has 12 dimensions. We shall test by initializing the genome of the GA with the centroids generated by the basic K-means result. A better starting point may help GAs to reproduce optimal result.

Acknowledgments. This work is partially supported by an ACE-Net post-doctoral fellowship and NSERC, Canada.

References

1. Asuncion, A., Newman, D.J.: UCI Machine Learning Repository. University of California, Irvine, CA (2007), http://www.ics.uci.edu/~mlearn/MLRepository.html
2. Latiff, N.M.A., Tsimenidis, C.C., Sharif, B.S.: Performance Comparison of Optimization Algorithms for Clustering in Wireless Sensor Networks (2007)
3. Lingras, P.: Applications of Rough Set Based K-Means, Kohonen SOM, GA Clustering. In: Peters, J.F., Skowron, A., Marek, V.W., Orłowska, E., Słowiński, R., Ziarko, W.P. (eds.) Transactions on Rough Sets VII. LNCS, vol. 4400, pp. 120–139. Springer, Heidelberg (2007)
4. Lingras, P., Chen, M., Miao, D.: Rough Multi-category Decision Theoretic Framework. In: Wang, G., Li, T., Grzymala-Busse, J.W., Miao, D., Skowron, A., Yao, Y. (eds.) RSKT 2008. LNCS (LNAI), vol. 5009, pp. 676–683. Springer, Heidelberg (2008)
5. Małyszkoa, D., Wierzchoń, Sławomir, T.: Standard and Genetic k-means Clustering Techniques in Image Segmentation (2007)