Genetic-guided Model-based Clustering Algorithms*

Hui-Dong Jin Kwong-Sak Leung
Department of Computer Sci. & Eng.
The Chinese University of Hong Kong
Shatin, Hong Kong

Man-Leung Wong
Department of Computing and Decision Sciences
Lingnan University
Tuen Mun, Hong Kong

Abstract *Clustering, or the unsupervised classification of data items into clusters, can reveal some intrinsic structures among data. The intrinsic structures, like the number of clusters, are one of the key issues in data mining and other exploratory data analyses. In this paper, we propose genetic-guided model-based clustering techniques to determine the optimal number of clusters and the characteristics of these clusters automatically. The model-based clustering techniques are used to describe the clusters and tune their descriptions locally, while genetic algorithms lead the search to some promising solutions. Several clustering-specific genetic operators are developed successfully to enhance the search procedure, as confirmed by the simulation results. The simulation results, both on synthetic and real-life data sets, demonstrate the better performance of the proposed clustering techniques over two widely-used model-based clustering algorithms.*

Keywords: Genetic algorithm, mixture model, expectation maximization algorithm, clustering analysis, number of clusters

1 Introduction

Clustering is the unsupervised classification of data items (observations or feature vectors) into meaningful groups (clusters) based on similarity. It can reveal some intrinsic structures, like the number of clusters and "natural" clusters among the data set, when no prior information is available other than the observed values. Clustering analysis is very useful in data mining [2, 4] and Web mining [12]. This appeals to researchers from many disciplines [10]. They have produced a rich assortment of clustering methods, like graph-based [11], model-based [2, 3] , genetic-guided [9], distance-based approaches [7].

Genetic Algorithms (GAs), motivated by natural evolution, maintain a population of solutions and make use of genetic operators to obtain the globally optimal solution [8]. This powerful optimization technique has been successfully combined with other clustering approaches to conduct exploratory data analysis. For example, the hybridization with K-means [7] or Fuzzy logic [9] has led to some successful results. However, it weakens their attraction greatly that users have to predefine the number of clusters in advance.

In theory, model-based clustering approaches are able to determine the optimal number of clusters automatically. They describe the data set with a mixture model, and a lot of theoretical results from the statistics field can be utilized to measure the match between the mixture model and the data set [1, 2, 6, 13]. The common procedure to select the optimal number of clusters is based on the enumeration strategy [1, 6, 13]. Given the number of clusters, it uses a local search algorithm, say, the Expectation Maximization (EM) algorithm, to find a good description for the data set, then these resulting models with different numbers of clusters compete with one another based on certain criterion. It is referred to as the *enumeration model-based clustering algorithm* (EnumEM) hereafter. The strategy wastes much computation time on the mixture models with an inappropriate number of clusters. As an alternative, some promising numbers of clusters may be driven from the clustering results during the running before invoking the EM algorithm [2]. Both of them suffer the

*The work was partially supported by RGC Grant CUHK 4161/97E of Hong Kong.

local search property of the EM algorithm.

By introducing a global search mechanism into model-based clustering techniques, we can improve the effectiveness to determine the number of clusters. The global search mechanism, say, GAs, can be used to both explore the natural clusters among data and determine the optimal number of clusters automatically. This distinguishes the genetic-guided model-based clustering approaches from most of the previous research work on GA-guided clustering algorithms [7, 9].

The remaining paper is organized as follows. In section 2, we outline some model-based clustering techniques involved. The genetic-guided model-based clustering approaches are proposed in section 3, followed by several specific genetic operators. Simulation results are given in section 4, and compared with EnumEM and AutoClass [2]. In section 5, we conclude the paper.

2 Model-based Clustering Analysis

Given a data set $x = (x_1, x_2, ..., x_N)$, the model-based clustering techniques assume that each data item x_i is drawn from a mixture model Φ with the density:

$$P(x_i|\Phi) = \sum_{k=1}^{K} p_k \phi(x_i|\theta_k). \tag{1}$$

Here K is the number of clusters, p_k is the mixing proportion (or, mixture weight) for the k^{th} cluster ($0 < p_k < 1$ for all k=1, ..., K and $\sum_{k=1}^{K} p_k = 1$), $\phi(x_i|\theta_k)$ is the corresponding density for cluster k and θ_k denotes the parameters involved.

This paper concentrates on the case where $\phi(x_i|\theta_k)$ is the multivariate normal (Gaussian) distribution, a model that has been used with considerable success [2, 5]. In this instance, the parameter θ_k consists of a mean vector μ_k and a covariance matrix Σ_k. The density is of the form

$$\phi(x_i|\theta_k) = \frac{\exp\left\{-\frac{1}{2}(x_i-\mu_k)^T\Sigma_k^{-1}(x_i-\mu_k)\right\}}{(2\pi)^{\frac{D}{2}}\,|\Sigma_k|^{\frac{1}{2}}} \tag{2}$$

where D is the dimensionality of the data items.

Given a mixture model Φ, we can get the membership probability $t_{ik} = \frac{p_k\phi(x_i|\theta_k)}{P(x_i|\Phi)}$ for the data item x_i. Then, we can get crisp classification by assigning the data items x_i to cluster k if $k = \arg\max_j\{t_{ij}\}$. Thus, a mixture model Φ can be viewed as a solution for the clustering analysis. We outline some techniques to find a good mixture model below.

2.1 The EM Algorithm

The commonly used *maximum log-likelihood* measures the description accuracy of a mixture model on the data set, defined as follows:

$$L_M(\Phi) = \sum_{i=1}^{N}[\log P(x_i|\Phi)] \tag{3}$$

The Expectation Maximization (EM) algorithm is one of the most famous model-based clustering algorithms [2, 3, 6, 13], given as follows.

The EM algorithm

1. Fixing the number of clusters K, initialize the parameters in the mixture model: p_k^j, μ_k^j and Σ_k^j ($k = 1, ..., K$), and set the current iteration $j = 0$.

2. **E-Step**: Given the mixture model parameters, compute t_{ik}^j:

$$t_{ik}^j = \frac{p_k^j\phi(x_i|u_k^j,\Sigma_k^j)}{\sum_{l=1}^{K} p_l^j\phi(x_i|u_l^j,\Sigma_l^j)} \tag{4}$$

3. **M-step**: Given t_{ik}^j, update the mixture model parameters for $k = 1, ..., K$:

$$p_k^{j+1} = \frac{1}{N}\sum_{i=1}^{N} t_{ik}^j \tag{5}$$

$$\mu_k^{j+1} = \frac{\sum_{i=1}^{N} t_{ik}^j x_i}{N\cdot p_k^{j+1}} \tag{6}$$

$$\Sigma_k^{j+1} = \frac{\sum_{i=1}^{N} t_{ik}^j (x_i-\mu_k^j)(x_i-\mu_k^j)^T}{N\cdot p_k^{j+1}} \tag{7}$$

4. If $\left|L_M(\Phi^{j+1}) - L_M(\Phi^j)\right| \geq \varepsilon$, set $j = j+1$ and go to step 2. Here ε is a small positive number.

As a greedy algorithm, EM increases the maximum log-likelihood $L_M(\Phi)$, and converges to a near optimal solution with high log-likelihood

value, but there is no guarantee to find the optimal one. Furthermore, the convergence rate may be very slow if the clusters are not well separated or the number of clusters is not properly predefined [5].

2.2 Hierarchical Agglomerative Clustering

Hierarchical Agglomerative Clustering (HAC) is an iterative procedure in which "optimal" pairs of clusters are successively merged. In the model-based HAC algorithm, a pair of clusters with the least loss of the *maximum classification log-likelihood* is chosen to agglomerate at each stage [5]. The classification log-likelihood is calculated according to:

$$L_C(\Phi) = \sum_{k=1}^{K} \sum_{x_i \in C_k} \phi(x_i|\theta_k) \qquad (8)$$

where C_k indicates the k^{th} cluster.

The time and the memory complexities of hierarchical agglomerative clustering algorithms depend quadratically on the number of components in the initial partition, which is usually a set of singleton clusters. Thus, it is impractical to process large data sets [5].

2.3 Model Selection

By balancing the model accuracy and the complexity, various criteria have been proposed to measure a model's suitability [1, 2, 13]. A classic criterion is the *Bayesian Information Criterion* (BIC). It is defined by:

$$BIC(\Phi) = -2 \cdot L_M(\Phi) + \upsilon(\Phi) \log N \qquad (9)$$

where $\upsilon(\Phi)$ is the number of free parameters in the mixture model Φ. Some simulation results have shown its good performance in practice [1, 2].

Based on this criterion, the widely used approach, EnumEM, works as follows. Given the number of clusters, the best maximum log-likelihood is estimated by invoking the EM algorithm several times with random initialization. Then, the BIC values for all possible K compete with one another. The model with the minimal BIC value is chosen to determine the number of clusters in the data set [1, 13]. Due to the local search property of the EM algorithm and little communication during running, the EnumEM algorithm does not work well, especially on complicated data sets.

AutoClass selects models using an "Occam Factor", which implying that Bayesian parameter priori can somehow prevent the over fitting [2]. During the running, a new promising number of cluster may be specified heuristically before invoking the EM algorithm. However, this strategy also suffers the local search property of the EM algorithm.

3 Genetic-guided Model-based Clustering Analysis

For the last decade there has been a growing interest in evolutionary algorithms that are based on Darwin's theory of evolution (Survival of the fittest). One of the implementations is Genetic Algorithms (GAs) [8]. GAs maintain a population of solutions and manipulate them with several genetic operators including mutation, crossover, and selection. These operators model some natural phenomena: genetic inheritance and Darwinian strife for survive. The most significant advantages of GAs are the flexibility and adaptability to the task on hand, the robust and global search characteristics. Thus they are often employed to handle inherently hard problems. GAs have been widely used in clustering analysis [7, 9]. These hybridization algorithms have to specify the number of clusters in advance, which greatly impacts their utility in real world problems.

By combining GAs with several model-based clustering techniques, we propose the genetic-guided model-based clustering analysis techniques. The global search capability of GAs guides the algorithms to focus on the promising mixture models and avoid the search for models with an inappropriate number of clusters as early as possible. The algorithms are outlined in Fig. 1 in which four different genetic operators are developed to enhance the performance. Different combinations of these operators can lead to different clustering algorithms. For convenience, we call the one with all the genetic operators GAXEM, and the one without the HAC crossover operator GAEM. These genetic operators are described below in details.

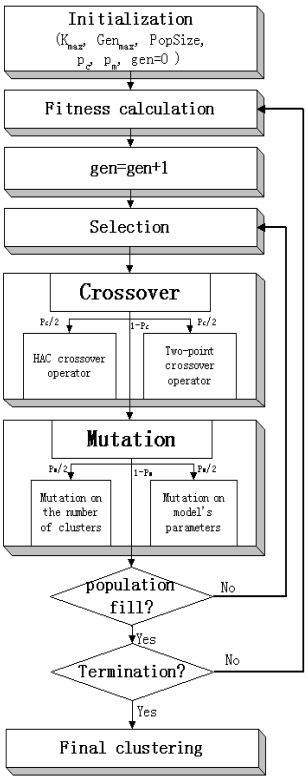

Figure 1: The flow chart of genetic-guided model-based clustering algorithms.

3.1 Representation and Fitness Calculation

In GAs, each mixture model is directly coded as a chromosome to represent a clustering solution: the first gene is the number of clusters, followed by genes representing the parameters for the first cluster, the second cluster, and so on. The parameters for a single cluster include the mixing proportion, the mean vector and the covariance matrix for the first cluster. So, the length of the chromosome is varying and depends on the number of clusters in the model.

The calculation of the fitness is based on an *n-iteration EM algorithm*. Starting from the model represented by a chromosome, EM runs the E-step and the M-step for n iterations. The updated model will replace the old one, and its maximum log-likelihood is used to calculate the BIC value according to Eq. (9). We use a selection operator for minimization. The less the BIC is, the more probable the chromosome is choosen. In our implementation, the number of iterations n increases linearly with the generation number.

3.2 Crossover Operators

There are two crossover operators as shown in Fig. 1. The first one, often used in GAs, is the two-point crossover operator. It exchanges the parameter values for some clusters between the two random crossover points.

The other one, a HAC crossover operator, is derived from the model-based Hierarchical Agglomerative Clustering (HAC) algorithm [5]. Two parent mixture models firstly merge into one new mixture model by appending one model behind the other followed by adjusting the number of clusters and the mixture proportions. Starting from this initial partition we can alleviate the significant overhead associated with the HAC algorithm. Then a pair of clusters with the least loss of the maximum log-likelihood will be chosen and agglomerated into a new cluster iteratively. The procedure stops when the BIC value reaches the miminum, which can lead to an optimal number of clusters.

Because the accurate calculation of the loss of the maximum log-likelihood is rather time-consuming, we just approximate it based on the assumption that $p_k\phi(x_k|\theta_k) >> p_s\phi(x_k|\theta_s)$ $(s \neq k)$ as x_k belongs to the k^{th} cluster C_k. That is, the maximum classification log-likelihood approximates the maximum log-likelihood. Then we has

$$\begin{aligned} L_M(\Phi) &\approx \sum_{k=1}^{K}\sum_{x_i\in C_k}\log\left(p_k\phi(x_i|\theta_k)\right) \qquad (10)\\ &= N\sum_{k=1}^{K}p_k\log p_k - \frac{1}{2}N\sum_{k=1}^{K}p_k\log|\Sigma_k| \\ &\quad -\frac{ND}{2}\log(2\pi) - \frac{1}{2}ND \end{aligned}$$

During the deduction, we use

$$\Sigma_k \approx \sum_{x_i\in C_k}\frac{\left\{(x_i-\mu_k)(x_i-\mu_k)^T\right\}}{Np_k}$$

and

$$\sum_{x_i\in C_k}\left\{(x_i-\mu_k)^T\Sigma_k^{-1}(x_i-\mu_k)\right\} = Np_kD.$$

If clusters l and j merge into cluster r, we have

$$\begin{aligned} &\Delta L_M(K) = L_M(\Phi_K) - L_M(\Phi_{K-1}) \qquad (11)\\ &\approx N\left(p_l\log p_l + p_j\log p_j - p_r\log p_r\right)\\ &-\frac{N}{2}\left(p_l\log|\Sigma_l| + p_j\log|\Sigma_j| - p_r\log|\Sigma_r|\right). \end{aligned}$$

As the number of free parameters for every cluster is fixed, say, F, we can reformulate $BIC(\Phi_K)$ and get

$$\begin{aligned} \Delta BIC(K) &= BIC(\Phi_K) - BIC(\Phi_{k-1}) \\ &\approx -2\Delta L_M(K) + F \log N. \end{aligned}$$

In order to minimize BIC within the HAC agglomeration procedure, we should keep $\Delta BIC(K) \geq 0$. That is

$$\Delta L_M(K) \leq \frac{F}{2} \log N. \tag{12}$$

Thus, we have a simple termination criterion for the HAC crossover operator to end with a good number of clusters.

Now let us derive the update formulae for the model parameters during the agglomeration procedure. We assume that, if cluster r agglomerates from clusters j and l, all their membership probabilities are agglomerated. So

$$u_r = \sum_{i=1}^{N} \frac{(t_{ij} + t_{il})x_i}{N(p_j + p_l)} = \frac{p_j \mu_j + p_l \mu_l}{p_j + p_l} \tag{13}$$

and

$$\Sigma_r = \frac{p_j \Sigma_j + p_l \Sigma_l}{p_j + p_l} + \frac{p_j p_l}{(p_j + p_l)^2} (\mu_j - \mu_l)(\mu_j - \mu_l)^T \tag{14}$$

It is worth noting that the HAC crossover operator does not need to access the data set again. Thus, the operator is suitable for large scale data sets.

3.3 Mutation Operators

The proposed GAXEM and GAEM have two mutation operators to introduce certain diversity into the population of chromosomes.

One is the mutation operator which just acts on the parameters of the model. It randomly selects a cluster, say k, and a data item, say x_i. Then the mean vector μ_k is moved to x_i with a small step. The covariance matrix Σ_k is adjusted by the covariance between μ_k and x_i. It is formulated as follows:

$$\begin{aligned} \mu_k' &= (1-\alpha)\mu_k + \alpha x_i \\ \Sigma_k' &= (1-\beta)\Sigma_k + \beta(x_i - \mu_k)(x_i - \mu_k)^T \end{aligned} \tag{15}$$

Here α and β are two small positive numbers.

The other mutation operator can mutate the number of clusters K. It first generates a new number K' around K. If $K' < K$, then the operator randomly selects K' clusters to form a new chromosome. Otherwise, it selects some data vectors as the mean vectors and the identity matrices as the covariance matrices to form some additional clusters. This mutation operator enables jumping from one mixture model to another, and favors the global convergence of GAXEM and GAEM.

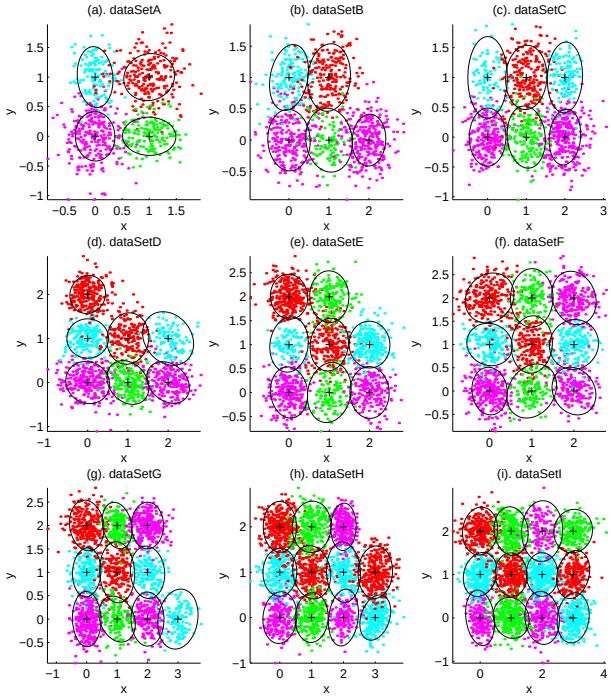

Figure 2: The 9 synthetic data sets and their original mixture models. The data items are indicated by dots. The center '+' and its corresponding solid ellipse indicate the mean and the contour of a component distribution.

4 Simulation Results

We illustrate the performance of the proposed GAXEM and GAEM algorithms on various data sets and compare them with EnumEM and AutoClass. We also make a comparison between GAXEM and GAEM to clarify the significance of the HAC crossover operator. All algorithms were implemented with MATLAB except that AutoClass was used its C version (http://ic-www.arc.nasa.gov/ic/projects/bayes-group), and executed on Sun Ultra 5/270. GAXEM and GAEM employed the elitism strategy. If the best fitness did not change for 5 generations, they were terminated.

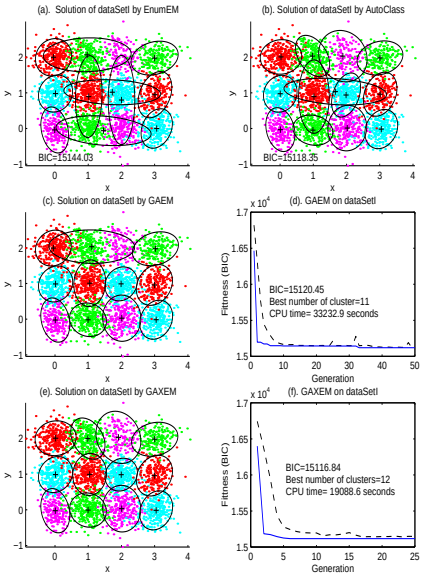

Figure 3: Typical results for dataSetI obtained by EnumEM, AutoClass, GAEM and GAXEM. In (a), (b), (c) and (e), the center '+' and its corresponding solid ellipse indicate the center and the contour of a component distribution generated. The dotted lines and the solid ones in (d) and (f) shows the mean and the minimal BIC values of the algorithms respectively.

To make it fair to compare with EnumEM and AutoClass, the population size was set as $5 \times N^{\frac{1}{3}}$. Here the number $N^{\frac{1}{3}}$ was the upper bound for the numbers of clusters, and the EM algorithm was invoked 5 times for every mixture model with a given number of clusters in EnumEM [6, 13]. Based on some preliminary experiments, the crossover and the mutation probability, respectively, were set to 0.99 and 0.05. These two mutation and two crossover operators were invoked randomly.

The first set of simulations has been conducted on 9 synthetic data sets. These 9 data sets are depicted in Fig. 2. Their center of the component distribution is located on the 2-dimensional grid and the covariance matrices are generated randomly around $0.1 * I_2$, where I_2 is the 2 by 2 identity matrix. We sample about 200 data items from each cluster for the first 7 data sets and 250 for the last two data sets.

The simulation results are summarized in Table 1 based on 10 independent runs. GAXEM can determine the optimal number of clusters more frequently than the others except for dataSetF. For example, EnumEM, AutoClass and GAEM succeed 2, 4 and 3 times, respectively, to detect 12 clusters among dataSetI within 10 runs, while GAXEM does 7 times. On average, EnumEM, AutoClass, GAEM and GAXEM respectively succeed 3.8, 7.1, 6.3 and 8.3 times within 10 runs. Similar situation happens on the other measurement: accuracy. It is defined to measure the match between two classifications C and C' by

$$Accuracy(C, C') = 1 - \frac{WeiEn(C, C') + WeiEn(C', C)}{2}$$

and $WeiEn(C, C') = \sum_{k=1}^{K} p_k \left[-\sum_{i=1}^{K'} q_{ki'} \log q_{ki'} \right]$,

where $q_{ki'}$ is the ratio of data items in cluster k of classification C assigned to cluster i' of classification C'. The accuracy value reaches the maximum 1 as two classifications are identical. Due to the ill-separated data sets, the accuracy values in Table 1 with respect to the original classification are hard to approach 1. GAXEM generates better classification with higher accuracy value than EnumEM, AutoClass and GAEM for all tested data sets except dataSetF. GAEM does better than EnumEM for all data sets except dataSetH. Fig. 3 gives some typical results and the running behaviors of GAEM and GAXEM. In Fig. 3(e), we can see that GAXEM generates a mixture model similar to the original one as shown in Fig. 2(i). As shown in Fig. 3(b), AutoClass is apt to generate more clusters than the optimal. In summary, the solutions obtained by GAXEM are better than those by the others. GAXEM performs better

Table 1: The simulation results on 9 synthetic data sets of EnumEM, AutoClass, GAEM and GAXEM. 'K' indicates the optimal number of clusters in the data set, 'Accu' the average accuracy value (%) and 'Suc' the successful trials on finding the optimal number of clusters within 10 runs. The unit for the average execution time 'Time' is second.

			EnumEM			AutoClass			GAEM			GAXEM		
Data set	K	N	Accu(%)	Suc	Time	Accu(%)	Suc	Time	Accu(%)	Suc	Time	Accu(%)	Suc	Time
dataSetA	4	800	56.8	8	1799	63.5	10	435	58.4	8	1079	63.5	10	1374
dataSetB	5	1000	51.4	4	2581	53.7	10	785	52.8	7	2096	53.4	9	3450
dataSetC	6	1200	42.6	5	2184	46.4	7	1031	47.9	8	3115	48.3	9	4385
dataSetD	7	1400	42.8	2	3135	58.6	8	1252	59.5	8	4803	63.7	10	4585
dataSetE	8	1600	62.0	4	3224	63.7	8	1543	64.2	7	4262	64.7	9	7380
dataSetF	9	1800	53.9	3	3318	60.2	9	1691	58.8	7	7399	58.6	7	7992
dataSetG	10	2000	54.0	3	4369	55.2	4	2158	56.9	5	9806	59.2	8	12921
dataSetH	11	2750	45.2	3	8570	47.1	4	2975	44.7	3	21871	52.9	6	26732
dataSetI	12	3000	37.5	2	9149	43.6	4	3763	50.3	3	29487	51.4	7	30296
Average			49.6	3.8	4259	54.7	7.1	1737	54.8	6.2	9324	57.6	8.3	12757

than GAEM, which confirms the significant role of the proposed HAC crossover operator. On the other hand, although GAXEM and GAEM need longer computation time, their computation time grows in a similar way as that of EnumEM. And, their computation time is within 10 times of that of AutoClass. AutoClass runs faster partially because it is coded in C programming language [2].

Our second set of simulations has been conducted on several real-life data sets from the UCI machine learning repository (www.sgi.com/Technology/mlc/db). The simulation results are summarized in Table 2. GAXEM can determine the optimal number of clusters in most simulations. On average, EnumEM, AutoClass and GAXEM respectively succeed 6.4, 4.4 and 8.6 times within 10 runs. Normally, GAXEM generates better solutions with lower BIC values than EnumEM and AutoClass for all 5 data sets. Here the BIC value for AutoClass is calculated based on the mixture models generated. Similar to the first set of experiments, AutoClass is apt to generate more clusters than the optimal. Especially, it failed to detect the five clusters for the data set 'sleep' within 10 runs, for which both EnumEM and GAXEM can often determine the five clusters correctly. This probably is because the "Occam Factor" in AutoClass [2] works on data sets different from the BIC in GAXEM. On the other hand, the execution time of GAXEM is longer than that of the other algorithms, but it is not significantly different from that.

In summary, GAXEM can determine the optimal number of clusters more frequently than EnumEM, AutoClass and GAEM with slightly longer execution time. GAEM, similar to AutoClass, performs better than EnumEM. GAXEM outperforms GAEM which shows the significant role of the proposed genetic operator.

Table 2: The simulation results on 5 real world data sets generated by EnumEM, AutoClass and GAXEM. 'K' indicates the optimal number of clusters in the data set, 'N' is the total number of data items, 'Attributes' indicates the attributes used in the simulation. 'Accu' the average accuracy value (%) and 'Suc' the successful trials on finding the optimal number of clusters within 10 runs. The unit for the average execution time 'Time' is second.

data set				EnumEM				AutoClass				GAXEM			
name	*K*	*N*	Attributes	*BIC*	Accu(%)	Time	Suc	*BIC*	Accu(%)	Time	Suc	*BIC*	Accu(%)	Time	Suc
diabetes	3	145	1,2,3	4762	60.4	81	9	4766	60.2	68	9	4770	60.7	113	10
thyroid	3	215	1,2,3,4,5	4948	74.0	104	6	4921	79.7	108	7	4810	83.0	203	8
iris	3	150	1,3,4	514	90.6	88	7	573	62.1	87	2	510	91.2	124	9
liver	2	345	1,2,3,4	1059	38.5	485	6	1223	36.9	264	4	1009	40.8	570	9
sleep	5	2500	5,7,8,10	38932	45.7	3045	4	42233	38.4	3481	0	37994	50.2	8793	7
Average				10043	61.8	761	6.4	10743	55.5	802	4.4	9819	65.2	1961	8.6

5 Conclusion

In this paper, we have proposed the new genetic-guided model-based clustering algorithms, GAXEM and GAEM. Besides finding the good clustering among data sets, they can determine the optimal number of clusters automatically. Based on the model-based clustering techniques, we have established several novel genetic operators to integrate the evolutionary mechanism with problem-specific techniques to improve its performance. The simulations both on synthetic and real-life data sets have illustrated that the proposed algorithms could determine the optimal number of cluster. Moreover, GAXEM and GAEM perform better than the widely used model-based clustering algorithm EnumEM, and GAXEM performs better than AutoClass in the preliminary experiments. Simulation results have also substantiated the significance of these proposed genetic operators. In the further, we will extend the proposed algorithms to handle the data sets with different kinds of attributes like AutoClass. The scalability of the proposed algorithms for large scale data sets is subject to our another research direction.

References

[1] C. Biernachi, G. Celeux, and G. Govaert. An improvement of the NEC criterion for assessing the number of clusters in a mixture model. *Pattern Recognition Letters*, 20:267–272, 1999.

[2] P. Cheeseman and J. Stutz. Bayesian classification (AutoClass): Theory and results. In Fayyad et al. [4], pages 153–180.

[3] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum-likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society Series B*, 1977.

[4] U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors. *Advances in Knowledge Discovery and Data Mining*. AAAI/MIT Press, 1996.

[5] C. Fraley. Algorithms for model-based Gaussian hierarchical clustering. *SIAM Journal on Scientific Computing*, 20(1):270–281, Jan. 1999.

[6] C. Fraley and A. E. Raftery. How many clusters? Which clustering method? Answers via model-based cluster analysis. *Computer Journal*, 41:578–588, 1998.

[7] P. Franti. Genetic algorithm with deterministic crossover for vector quantization. *Pattern Recognition Letters*, 21(1):61–68, 2000.

[8] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Pub. Co., 1989.

[9] L. O. Hall, I. B. Özyurt, and J. C. Bezdek. Clustering with a genetically optimized approach. *IEEE Transactions on Evolutionary Computation*, 3(2):103–112, 1999.

[10] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(3):264–323, September 1999.

[11] George Karypis, Eui-Hong (Sam) Han, and Vipin Kumar. Chameleon: Hierarchical clustering using dynamic modeling. *Computer*, 32(8):68–75, August 1999.

[12] Olfa Nasraoui, Raghu Krishnapuram, and Anupam Joshi. Relational clustering based on a new robust estimator with application to web mining. In *18th International Conference of the North American Fuzzy Information, 1999*, pages 705–709, 1999.

[13] L. Xu. Bayesian Ying-Yang machine, clustering and number of clusters. *Pattern Recognition Letters*, pages 1167–1178, Nov. 1997.