International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:1, No:5, 2007

A Genetic Algorithm for Clustering on Image Data

Qin Ding and Jim Gasvoda

Abstract**—** Clustering is the process of subdividing an input data set into a desired number of subgroups so that members of the same subgroup are similar and members of different subgroups have diverse properties. Many heuristic algorithms have been applied to the clustering problem, which is known to be NP Hard. Genetic algorithms have been used in a wide variety of fields to perform clustering, however, the technique normally has a long running time in terms of input set size. This paper proposes an efficient genetic algorithm for clustering on very large data sets, especially on image data sets. The genetic algorithm uses the most time efficient techniques along with preprocessing of the input data set. We test our algorithm on both artificial and real image data sets, both of which are of large size. The experimental results show that our algorithm outperforms the k-means algorithm in terms of running time as well as the quality of the clustering.

Keywords**—**Clustering, data mining, genetic algorithm, image data.

I. Introduction

THE task of grouping data points into clusters of "similar" items is a form of unsupervised learning that has application in many fields. For instance, current techniques used for machine vision require processing of digital information obtained from pixels. A very important step in this digital information processing is to group the data in some fashion so that patterns can be recognized. Clustering can be used for this task. In the medical field, clustering of data can be used to determine if a drug provides greater benefits to a certain group of patients. Grouping of information is used in the engineering field to determine what factors lead to the failure of a component in a system. And in marketing, data clustering can give a clearer picture of how to focus an advertising campaign to the proper audience.

This paper will discuss the use of Genetic Algorithms (GAs) for the task of clustering data. In particular, the application of GAs for clustering on very large data sets, such as image data sets, will be addressed. The running time for most clustering GAs becomes quite large as the size of the input data set increases. We propose an efficient genetic algorithm for clustering on very large image data sets.

Manuscript received August 12, 2004.

Q. Ding is with the Pennsylvania State University – Harrisburg, Middletown, PA 17057 USA (corresponding author to provide phone: 717-948-6636; fax: 717-948-6352; e-mail: qding@psu.edu).

J. Gasvoda is with the Pennsylvania State University – Harrisburg, Middletown, PA 17057 USA (e-mail: jmg289@psu.edu).

The paper is organized as follows. In section II, we review the clustering problem and genetic algorithms. In Section III, we detail our genetic algorithm for clustering on very large data sets. Experimental results on both artificial and real image data sets are given in Section IV. Section V concludes the paper.

II. Clustering problem and genetic algorithms

A. Clustering Problem

The diversity of applications for clustering has lead to many problem definitions. The objective of all clustering algorithms is to divide a set of data points into subsets so that the objects within a subset are similar to each other and objects that are in different subsets have diverse qualities [6], [8], [9]. The fact that there are many different methods used to quantify the similarity and diversity of data points leads to the many different variations of the problem. For our research, we defined the clustering problem as the task of dividing an input data set into a desired number of subgroups so that the Euclidean distance between each data point and its corresponding cluster center is minimized. This is a very common method of defining the clustering problem. The total of the distances of each point to its cluster center is known as the total distance measurement of the clustering and is calculated as shown in (1).

$$E = \sum_{k=1}^{K} \sum_{x \in Ck} \sum_{a=1}^{A} (x_a - mk_a)^2 \quad (1)$$

In this formula K is the number of clusters, x represents a data point, Ck represents cluster k, mk represents the mean of the cluster k, and A is the total number of attributes for a data point. This formula simply calculates the Euclidean distance of each point in the input data set to its cluster center. Minimizing the total distance measurement of a clustering leads to an optimal clustering solution. This definition, like all clustering definitions, requires finding an optimal collection of subsets for a group of data points. This class of problem is known to be NP-Hard. Work has been done to develop both approximate and exact solution algorithms for solving various clustering problems [1] but the solutions appear to be impractical, as either the number of data points in the input set or the number of clusters desired becomes large. As a result, there have been a wide variety of heuristic algorithms developed for the clustering problem. These algorithms do not

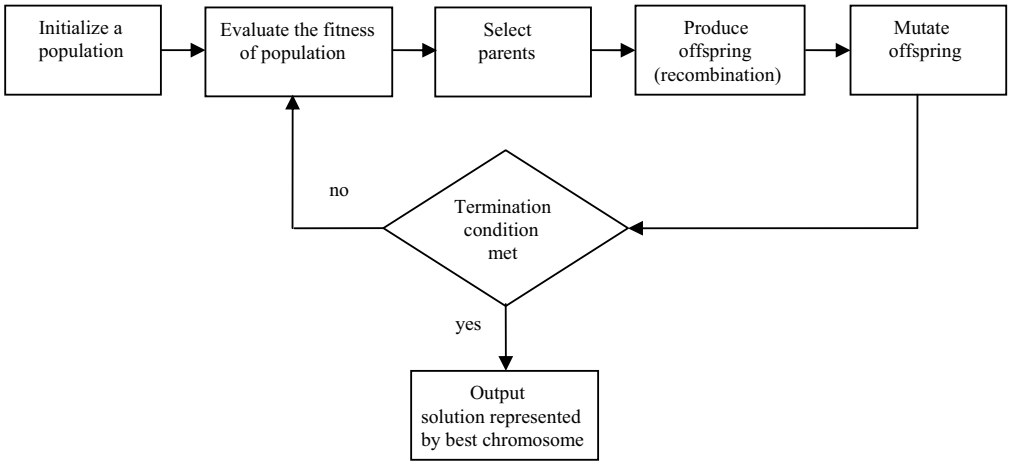


Fig. 1. Process of Genetic Algorithms

guarantee any quality in the solutions they find but they do run in polynomial time with respect to the number of objects in the input data set and the number of desired clusters.

B. Genetic Algorithms

Evolutionary Computation (EC) is a field of computer science that uses biological processes as a model for solving computer-based problems [2]. Genetic Algorithms (GAs), first proposed by John Holland in the 1960s, are a category of EC that use concepts derived from evolution. Proper application of a GA finds a balance between exploration and exploitation of a given optimization problem's search space. A good overview of how to design a GA is given in [11]. Fig. 1 shows the structure that is used by GAs. First, a population of chromosomes is created an initialized. These chromosomes each contain a collection of genes and each gene has a value (called an allele). A single chromosome is an encoded version of a solution to the problem that the GA is attempting to optimize. The GA performs exploration/exploitation of the problems search space by evolving the population of chromosomes through a series of generations. During each generation of the GA, parent chromosomes are selected from the population. These parent chromosomes are combined to form children chromosomes and then the child chromosomes are mutated. In a generational type GA, an entirely new population for each generation is formed by creating multiple child chromosomes. For a steady state GA, the child chromosomes are used to replace members of the current population but a new population is not formed during each generation.

A very important step in the GA is the selection of parents for the next generation of chromosomes. In order to provide a guided search, which is appropriate for the given optimization problem, the selection of parents needs to be based on the quality of the solution that their chromosomes represent. A property called fitness is used to quantify the quality of a given solution and a fitness function is used to calculate the fitness value of each chromosome in a given population before parent selection is made. A variety of different selection methods are used by GAs but they all use the principle that higher fit chromosomes are more likely to be chosen as parents. This fitness selection provides the GA direction for the search of an optimization problems search space.

III. A Genetic Algorithm For Clustering on image Data

A. Genetic Algorithms for Clustering Data

Using a GA to solve data clustering problems is not a new idea. GAs have been successfully implemented for various clustering problems using different chromosome encoding schemes and fitness functions. In [12] a GA is used to solve the clustering problem for a data set of geographical data. Each data point in the input data set is assigned a unique integer value from 1 to n, where n is the total number of data points in the input set. The chromosomes in a population contain one gene for each data point that is to be clustered and the allele values of the genes designate the assignment of all n points to the desired number of clusters. The total length of a chromosome is n. The fitness function used in the GA mimics the objective function of the k-means algorithm, which is shown in (1). The algorithm described in [3] uses a multi-step procedure. The authors refer to this procedure as a semi-supervised form of learning. A GA performs clustering on an input set of data objects so that supervised learning can be applied to predict class labels in the second step. The input for the GA is a set of data objects that have both numeric and label attributes and a desired number of clusters. The goal of the GA is to produce clusters of data objects that minimize cluster dispersion and are as pure as possible in relation to the label attributes. The GA uses a two component fitness function where the first component measures the within

cluster variance using a distance metric and the second component measures the similarity of the labeled attributes of the data objects using the GINI index. The encoding in [3] uses gene values to define the location of the cluster centers. An alternate encoding of chromosomes is used in [5]. This encoding uses medoid-based centers in which k input data points are chosen to be the centers of the corresponding k clusters. Each data point is assigned a unique number. The data point numbers are used in the chromosomes to designate the medoids for the encoded clustering. A novel idea of using variable length chromosomes is presented in [10]. The fitness function used is very similar to equation (1) and the encoding of the chromosome is the same as [12] where there are two genes to represent each cluster center in the 2D space.

B. A Genetic Algorithm for Clustering on Image Data

All of the algorithms discussed above become impractical as the input data set becomes very large. The encoding scheme of [12] requires one gene in the chromosome for each data point in the data set. Obviously, this encoding scheme cannot be used for very large data sets because the memory requirements to maintain a population of chromosomes would be restrictive. The long running times of each of the clustering GAs discussed above makes its application to very large data sets unrealistic. If the input data set contains one million objects with six attributes and a GA run involves 50,000 fitness evaluations, which is not uncommon, then there will be on the order of 3×10^{11} (1,000,000×50,000×6) calculations executed while performing the clustering task.

To solve this problem, we need to use efficient techniques, such as efficient encoding techniques, in the GA process. In addition, preprocessing the input data set can be a possible way to significantly reduce the execution time of clustering GAs for very large data sets. Preprocessing has been applied to clustering algorithms, other than GAs. The preprocessing results in a smaller data set can then be used as representation of the full input data set. Two ways of preprocessing are sampling and summarizing. Sampling of the input data set is straightforward. Summarizing the input data is a more complicated subject. Various algorithms have been devised to perform the task of summarizing data sets. Reference [4] provides a grid-based method of replacing a region of space containing a large number of points with a smaller number of representative points. The representative points contain attributes that summarize information about the whole set of points in the region of space.

The clustering algorithm that we designed for application on very large data sets, such as image data sets, is discussed below. We designed our GA with the intent of making it as fast as possible by choosing genetic algorithm techniques that are optimal in terms of quickness of execution. Our algorithm also uses data set preprocessing to reduce the running time.

1) Algorithm

The input to the clustering algorithm is a data set along with the desired number of clusters. The goal of the algorithm is to divide the input data set into the desired number of clusters so that the Euclidean distance between each data point and its corresponding cluster center is minimized.

A steady state GA was used. This steady state GA was chosen over the generational type of GA because in preliminary tests it was faster. The GA starts with an initial population of chromosomes and then the population is evolved through generations. During each generation two parents are selected and two child chromosomes are created using a recombination genetic operator. Each child is mutated and the most fit child is identified and used for replacement operator. The number of chromosomes in the population stays constant as the population is evolved through generations. Evolution is continued until a termination condition, in our case a given number of generations, is met.

The encoding technique used in our GA is similar to the one described in [3]. Fig. 2 illustrates how a sample gene is used to encode a clustering for a data set containing data points with two numeric attributes. The circles denote data points and the x's indicate the cluster centers that are specified in the encoded chromosome. In this encoding technique the cluster centers are encoded into the chromosome and the length of each chromosome is proportional to the number of attributes of a data point and also the number of desired clusters. As long as the desired number of clusters times the number of attributes is much less than the total number of input data points (which is normally true), this type of encoding is scalable for use with a very large data set. It is also a simple encoding technique that allows for quick decoding during fitness evaluation. For the encoding technique we used genes with real values, an alternative was to use binary values. The topic of using real values or binary equivalent representations is discussed in [7] where it is concluded that using real allele values in the genes resulted in a faster algorithm with nearly the same quality of solutions. We therefore choose to use chromosomes with real valued alleles.

As many as 50,000 fitness evaluations are common during one run of a GA. The impact that the fitness function has on the running time of the GA can easily be seen. The complexity of the fitness function, in terms of the number of calculations, must be kept very simple for any GA that is to be used to perform clustering on a very large data set. The fitness function we used for the GA is the same as the one shown in (1). Since the objective function of the k-means clustering algorithm is also equal to the equation in (1), it is easy to compare the results of the resulting GA to the k-means algorithm.

With the fitness function and encoding chosen, the next step was to specify the genetic operators for the GA. Roulette wheel selection was used to determine each parent for recombination. For recombination, one-point crossover with two parents and two offspring was used. Each allele in each offspring chromosome gene was mutated with probability of 7%. The mutation of an allele was accomplished by randomly picking a value from a normal distribution with mean of zero and standard deviation equal to MAX/40, where MAX is

the

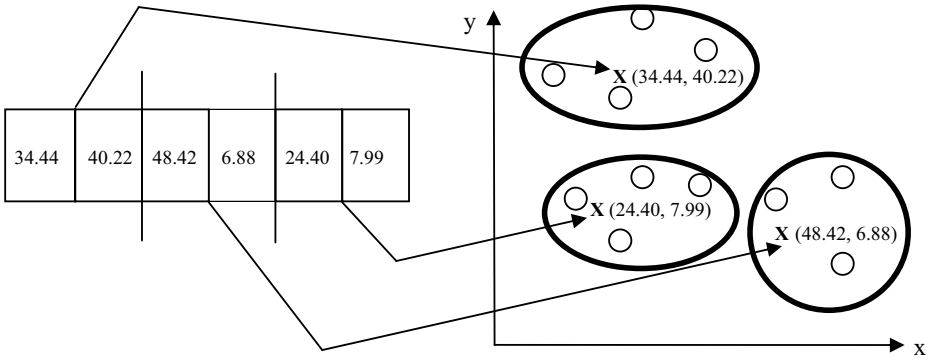


Fig. 2. Encoding in Genetic Algorithms

maximum attribute value of any data point in the original data set. The mutated value was then checked to make sure that it was no larger than MAX or smaller than MIN, where MIN is the smallest attribute value for any data point. If the value was greater than MAX or less than MIN the mutation was cancelled.

2) Preprocessing of Input Data Set

A very large input data set can be preprocessed to make a representative set that can be used by the algorithm for better time and space efficiency. We implemented two alternate preprocessing methods for our clustering algorithm. The first preprocessing method used random sampling to obtain a data set with fewer points. This reduced data set was then used in evaluating the fitness of the chromosomes. The second preprocessing method used summarization of the input data set and is based on the work presented in reference [4]. For this method, a grid is first constructed and then the input data set is applied to this grid. A single point location and corresponding weight is calculated for each region defined by the grid. The location of the representative point is chosen as the mean value of all the points in the region and the weight of the representative point is equal to the number of points that it replaces.

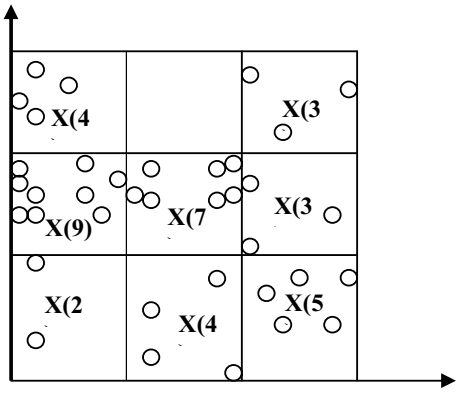


Fig. 3. Summarization of input data

Fig. 3 shows an example summarization of an input data set where each data point contains two attributes. The circles represent the data points and the Xs denote the representative points for each of the grid regions. The number in the parenthesis indicates the weight of the representative point for this example. A region of the grid with no points from the input data set has no representative data point. This grid process is extendable to input data sets with data points that have n attributes by using grid regions containing n dimensions.

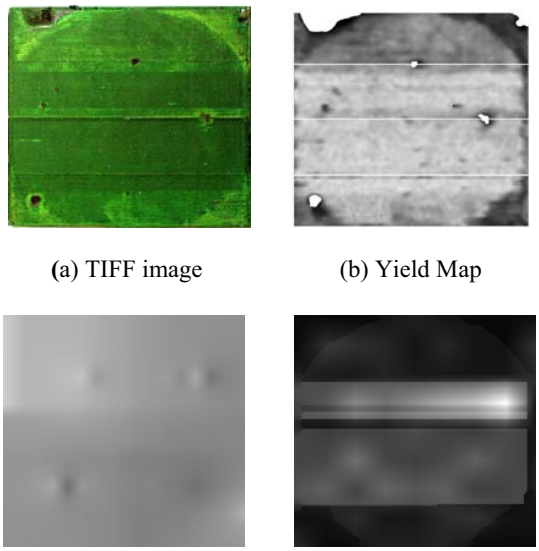
IV. Performance Analysis

A. Input Data Sets

To test the performance of our GA, artificial as well as real image data sets were used. Both the artificial data sets and real data sets contain six numerical attributes with values between 0 and 255. The artificially generated data sets were generated using a method of generation that is a modification to the one used in [5]. A data set contains n points with the points centered around k cluster centers. The k cluster centers are first determined by randomly and uniformly choosing each of the six attributes values from a range of 0 to 255. The minimum distance between any two cluster centers is then calculated, call this value D. To generate a data point a cluster center is chosen randomly. This cluster center is then used to calculate the six attribute values for the data point. Each attribute value for the data point is calculated by taking the attribute value of a cluster center and adding an offset to it. The offset is chosen randomly from a normal distribution with mean of 0 and standard deviation of D/r, where r is a variable that can be used to specify the tightness of the clusters. This process is repeated until the data set contains n data points and k clusters.

Each real data set contains a group of aerial photographs and associated ground data. The clustering problem is to group the pixels based on the attribute (also called band) values in those images. For example, Fig. 4 is a set of four

images about the Oakes area in North Dakota in 1997. The first image is an aerial photograph containing three bands, i.e. red, green and blue. The other three images contain synchronized soil moisture, nitrate, and yield values respectively. Each image is of the size 1320×1320. The 35MB file containing 1,700,000 records and 6 attributes is available at: http://www.midas.cs.ndsu.nodak.edu/~ding/images.



(a) TIFF image (b) Yield Map

(c) Moisture Map (d) Nitrate Map

Fig. 4. A Real Image Data Set

B. Results

We ran several tests to evaluate the performance of our clustering GA on very large image data sets. All of the tests were conducted on a PC with processor speed of 2.5 GHz and 512 MB of RAM running windows XP. The clustering quality and running time was determined for the clustering GA for a variety of input data sets and desired cluster sizes. The quality of the clustering was calculated as the sum of the Euclidean distance of each input data point to its corresponding cluster center. A lower distance measurement indicates a better quality clustering.

We tested our clustering GA using both forms of data preprocessing. For the clustering GA with summarization preprocessing (GA-SUM) the processed data set used to evaluate the fitness function was obtained using a grid system as previously explained. The grid was chosen such that each attribution range was divided into seven parts. A grid region was represented by a six-dimension cube, one dimension for each attribute of a data point in the data set. A representative point and corresponding weight was calculated for each region. For the clustering GA with random preprocessing (GA-RAN) the fitness of each chromosome was calculated by using a random sample of the input function. For comparison purposes the size of the sample set for each input data set was chosen so that the running times of the GA-SUM and GA-RAN were about equal. This was done in an attempt to see if one preprocessing method would provide a better quality of clustering given a set amount of time.

The GA-SUM and GA-RAN were tested using a combination of different input data sets. The k-means algorithm was also tested on the same input data sets to provide a performance benchmark. We implemented the k-means algorithm using open source software from the University of Tokyo, Institute of Medical Science, Human Genome Center (http://bonsai.ims.u-tokyo.ac.jp/).

Table I provides a summary of the results of the tests that we ran on the artificially generated data sets. For the artificial data set, calculation of the total distance of all the data points from their actual center is possible because the number of cluster centers and their location are known. This calculation provides a good benchmark for evaluation of our GA. The total distance of the k-means algorithm also provides a good benchmark for evaluating the quality of solutions found and is included in Table I. Each algorithm was run through 100 trials for a given input set and the average was computed for the running time and distance measurement.

TABLE I
RESULTS ON ARTIFICIAL DATA SET

Input Set	Algorithm	Running Time (sec)	Distance Measurement*
5 Centers 10000 Points	GA**	312	6.59×10^6
	GA-RAN	3	1.67×10^7
	GA-SUM	2	4.59×10^7
	k-means	.38	2.27×10^7
	Actual Centers	-	8.15×10^5
7 Centers 100000 Points	GA-RAN	22	1.23×10^8
	GA-SUM	20	1.55×10^8
	k-means	20	1.84×10^8
	Actual Centers	-	2.81×10^7
10 Centers 250000 Points	GA-RAN	49	5.15×10^8
	GA-SUM	50	6.07×10^8
	k-means	98	8.20×10^8
	Actual Centers	-	9.05×10^7

* a smaller distance measurement indicates a better solution
** the GA with no preprocessing was run for only 10 trials

Table II shows the results obtained from running the GA-RAN, GA-SUM and k-means algorithm on the real image data set. For this real image data set there is no notion of actual data cluster centers used to generate the data. The goal of running an algorithm on real image data sets is to choose a number of cluster centers to see if interesting and meaningful patterns can be obtained. Again, each algorithm was run through 100 trials on the data set with the average of the running time and distance measurement computed. The GA-

SUM and the GA-RAN both found better quality clusterings faster than the k-means algorithm for the two larger artificial data sets. They also outperformed the k-means algorithm for all cluster values on the real image data set. The test results indicate that the GA-RAN slightly outperforms the GA-SUM in terms of the quality of clustering that are found. For the artificial data set containing 10,000 data points, the GA with no input set preprocessing was run for 10 trials instead of 100 trials because of the longer running time. The result shows that some type of preprocessing is needed because the running time for the GA alone becomes very long as the input set size becomes large.

TABLE II
RESULTS ON REAL IMAGE DATA SET

Cluster Centers	Algorithm	Running Time (sec)	Distance Measurement
5	GA-RAN	89	2.27×10^9
	GA-SUM	91	2.87×10^9
	K-Means	193	3.02×10^9
7	GA-RAN	133	2.25×10^9
	GA-SUM	123	2.25×10^9
	K-Means	385	2.35×10^9
10	GA-RAN	241	1.72×10^9
	GA-SUM	226	1.74×10^9
	K-Means	524	1.85×10^9

V. CONCLUSIONS

Clustering is an important task with applications in many fields. Heuristic algorithms are used for this task in an attempt to provide acceptable results, both in terms of solution quality and running time, because all of the non-trivial clustering problem variations are NP-Hard. GAs have been applied to the clustering problem for many applications with some success as described in section III. For clustering on very large data sets, such as image data sets, the size of the associated databases makes it necessary to modify traditional GAs because of their slow running times. In this paper we proposed a steady GA algorithm with efficient encoding technique and GA operators along with input set preprocessing. Experimental results were promising. For input data sets with 100,000 points and larger, our GA provided better quality solutions faster than the k-means algorithm.

The results of our tests indicate that, given about the same amount of time to run, the GA-RAN provides slightly better quality solutions than the GA-SUM. The input data set characteristics, such as number of outliner points and tightness of data grouping determine which preprocessing technique is better. The summary preprocessing method that we implemented could be refined to prevent the creation of representative points for regions that contained less than a certain minimum threshold of points. This refinement would remove the negative effect that outlier points have on the clustering quality. It would also make the GA-SUM run faster because there would be fewer points in the processed data set.

REFERENCES

[1] P. K. Agarwal and C. M. Procopiuc, "Exact and approximation algorithms for clustering," in *Proceedings of the ninth annual ACM-SIAM symposium on Discrete algorithms*, 1998, pp. 658-667.
[2] P. J. Angeline, "Adaptive and self-adaptive evolutionary computations," *Computational Intelligence: A Dynamic System Perspective*, Piscataway, IEEE Press, 1995, pp. 152-163.
[3] A. Demiriz, K. P. Bennett, and M. J. Embrechts, "Semi-supervised clustering using genetic algorithms," R.P.I. Math Report No. 9901, Rensselaer Polytechnic Institute, 1999.
[4] W. DuMouchel, C. Volinsky, T. Johnson, C. Cortes, and D. Pregibon, "Squashing flat files flatter," in *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 1999, pp. 6-15.
[5] V. Estivill-Castro and A.T. Murray, "Spatial clustering for data mining with genetic algorithms," in *Proceedings of the International ICSC Symposium on Engineering of Intelligent Systems*, 1998, pp. 317-323.
[6] J. Grabmeier and A. Rudolph, "Techniques of cluster algorithms in data mining," *Data Mining and Knowledge Discover*, 6, 2002, pp. 303-360.
[7] L. O Hall, I. B. Ozyurt, , J. C. Bezdek, "Clustering with a genetically optimized approach," *IEEE Transactions on Evolutionary Computation*, 3(2), 1999, pp. 103-112.
[8] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann Publishers, 2000.
[9] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM Computing Surveys*, 31(3), 1999, pp. 264-323.
[10] C.-Y. Lee and E. K.Antonsson, "Dynamic partitional clustering using evolution strategies," in *Proceedings of the Third Asia Pacific Conference on Simulated Evolution and Learning*, 2000.
[11] M. Mitchell, *An Introduction to Genetic Algorithms*, MIT Press, 1998.
[12] M. Painho and F. Bação, "Using genetic algorithms in clustering problems," in *Proceedings of GeoComputation Conference*, 2000.

Qin Ding received her Ph. D. in computer science from North Dakota State University, Fargo, ND, USA, in 2002, M. S. and B. S., both in computer science, from Nanjing University, Nanjing, China, in 1991 and 1988 respectively.

She is currently an Assistant Professor in computer science at Pennsylvania State University – Harrisburg, Middletown, PA, USA. She was a Research Assistant in Computer Science Department at North Dakota State University from 1998 to 2002. Prior to that, she was a lecturer at Computer Science and Engineering Department at Hohai University, China. Her research interests include database, data mining, and bioinformatics.

Dr. Ding is a member of Association of Computing Machinery (ACM).

Jim Gasvoda received his B. E. in electrical engineering from Montana State University, Bozeman, MT, USA, in 1986. He is currently a graduate student in computer science at Pennsylvania State University – Harrisburg, Middletown, PA, USA.