A COMBINED MULTI-DIMENSIONAL DATA MODEL, SELF-ORGANIZING ALGORITHM AND GENETIC ORITHM FOR CLUSTER DISCOVERY IN DATA MINING

Hoàng Kiếm – Đỗ Phúc
Trường Đại Học Khoa Học Tự Nhiên
(Bài nhận ngày 06/10/1998)

ABSTRACT : Data mining is the discovery of patterns that may exist implicitly in a large database. In this paper, we study a combined model for cluster discovery. We build a multi dimensional data model (MDDM) and transform tuple of database into vector of MDDM then we use Kohonen's self-organizing algorithm to discover the potential clusters and Genetic Algorithm (GA) for validating these clusters. By combining MDDM with GA, we propose a heuristic for improving the efficiency of cluster discovery.
Keywords: cluster center, c-means algorithm, heuristic, genetic algorithm, self-organizing algorithm, multi-dimensional data model, optimization problem, vector quantization

1.INTRODUCTION

Cluster discovery is the process of grouping n objects into pre-defined c clusters based on a measure of similarity. From these clusters, we can build rules to characterize nature or to predict the behavior of object [8]. There are many traditional clustering algorithms but these algorithms are not designed for large data set [5,7].

When using clustering algorithm to data mining application, we have to solve the following problems:

- Number of clusters to be discovered
- Number of objects to be partitioned.
- Efficiency of clustering algorithm.

Our method was established to solve these problems. We build a combined MDDM and Kohonen's self-organizing algorithm for discovering the number of potential cluster centers.

For increasing the efficiency, we applied the random search approach that was proposed in [7] and use GA to implement this idea. To discover data clusters, we find a set of cluster centers [2,4,5]. We transform the clustering problem into an optimization problem as follows.

Clustering discovery is to find a set of cluster centers so that the sum of average distance between object and its cluster center is minimized and the sum of distance between cluster centers is maximized.

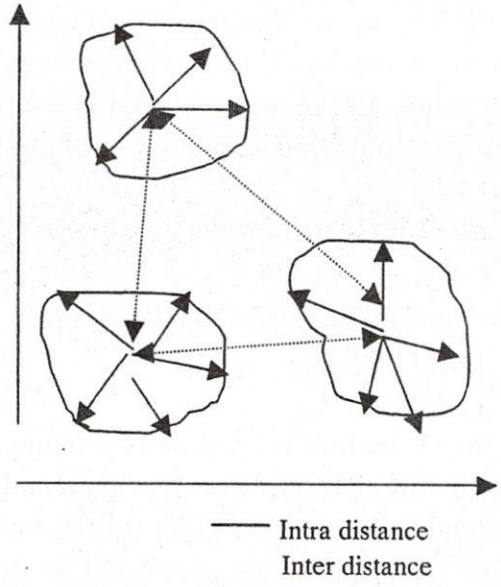

Figure 1: Cluster centers

2.TRANSFORMING TUPLE INTO VECTOR

We build a MDDM for transforming a tuple of database into a vector of MDDM and analyzing the distribution of vectors of MDDM [3]. In the following, we use only a two-dimensional model, but it is easy to expand this model to multi-dimensional model. Suppose we have a data table DB(X,Y), X and Y are numerical attributes, tuple (X,Y) is an input vector. Let DX, DY be the domain of attribute X ,Y of database DB (X,Y). We partition DX into intervals $D_{11},...,D_{1n}$ and DY into intervals $D_{21},...,D_{2m}$. We define MDDM as follows:

$\{D_{11},...,D_{1n}\}x\{D_{21},...,D_{2m}\}$----- > N

(X_1,X_2) ---------- > $f(X_1,X_2) \in N$

Where N is a natural number set. $\{D_{11},...,D_{1n}\}x\{D_{21},...,D_{2m}\}$ is the Cartesian product. One typical example of 2-dimensional MDDM is shown in table 1:

D25	0	0	11	21	18
D24	0	0	12	17	19
D23	0	0	10	8	9
D22	0	0	0	0	0
D21	0	0	0	0	0
	D11	D12	D13	D14	D15

Table 1: A typical two-dimensional MDDM with 5 intervals in dimension X1 and 5 intervals in dimension Y1

The value $f(X_1,X_2)$ in the above table is the number of tuples (X,Y) where value of X is in the interval X_1 and value of Y is in the interval X_2. (X_1,X_2) is the cell address.

For partitioning dimension into many intervals, we need to decide how many partitions there should be. As mentioned in 1, we use MDDM for building architecture of

Kohonen's self-organizing algorithm and controlling the direction of mutation operator of GA. For increasing the efficiency, we build MDDM based on the capacity of main memory of computer and the number of dimensions of vector. Let C be the capacity of main memory, S be the memory space for a cell value, L1,...,Ln be the number of intervals for dimension X1,...,Xn respectively then the following condition must be satisfied.

$$L1 \times ... \times Ln \times S < C$$

3. USING KOHONEN'S SELF- ORGANIZING ALGORITHM TO DISCOVER POTENTIAL CLUSTER CENTERS

Based on the above MDDM, we build a Kohonen's neural network and combine vector quantization (VQ) with the self organizing algorithm to discover the potential cluster centers [1]. Each cell of MDDM will be a weight vector w_{ij} of node (i,j) of the feature map, i =1..M, j = 1..N, i, j are the indices of the grid; M is the number of intervals in dimension X1 and N is the number of intervals in dimension X2.

The self-organizing algorithm can be summarized as follows:

1. Initialize all weight vectors to the coordinates of centers of MDDM cells.
2. Select the node with minimum distance dv to the input vector v(t).

$$dv(v,w_{icjc}) = \min \{ dv(v_i,w_{ij}) \}$$

3. Update weight vectors of nodes that lie within a nearest neighbor set of the node (i_c,j_c):

$$w_{ij}(t+1) = w_{ij}(t) + \alpha(t)(v(t)-w_{ij}(t))$$

For $i_c-N_c(t) <= i <= i_c+N_c(t)$ and

$j_c-N_c(t) <= j <= j_c+N_c(t)$

4. Update time t = t+1, add new input vector and goto (2)

In the above algorithm, $\alpha(t)$ is a gain ratio ($0 <= \alpha(t) <= 1$) and $N_c(t)$ is the radius of neighbor set. $N_c(t)$ and $\alpha(t)$ are decreased monotonically with time.

Each weight vector is assigned to variable c_s. This variable will hold the number of adapted times of this weight vector during Kohonen's self-organizing learning.

After using self-organizing algorithm, we may have data in table of table 2:

X1	X2	# OF ADAPTED TIMES
0.4	0.7	1507
6.5	4.2	4500
9.7	8.4	4600
...	...	.....

Table 2: Potential cluster centers

In table 2, (X1,X2) is the coordinate of weight vector, the third column is the number of adapted times of the weight vector during the self-organizing process. The self-organizing algorithm that was presented above may have poor quantization performance since the learning may not necessary lead to global or local minimum as defined in conventional VQ methods. After stopping the above algorithm ($\alpha(t) = 0$ or $N_c(t)=0$), we use only weight vectors that the number of adapted times are greater than a Threshold value. These vectors are the potential cluster centers for the GA based validation process.

4. USING GA FOR CLUSTER VALIDATION

4.1.Principle of Genetic Algorithm

The basic idea of a GA is do what the nature does. GA uses the natural evolution for solving the optimization problems in real life. From the initial set of solutions, the evolution will be repeated in many steps (evolving) under the action of the operators, the new set will be formed with the better solutions and finally the last solution will be the best solutions that best fit the optimization process. The structure of GA can be summarized in the following procedure [6,9]:

```
Procedure GA;
  Begin
t = 0
Initialize P(t);
Evaluate P(t);
While not terminate-condition do
        Begin
          t=t+1
          Select P(t) from P(t-1);
          Recombine P(t);
          Evaluate P(t);
        End
  End
```

4.2.Stating the cluster validation as an optimization problem

Let D_v be the input data set. We want to partition this set into K clusters, each cluster will be represented by a cluster center w_k. Let D_w be a set of K cluster centers. Vector v_i belongs to cluster $C(w_s)$ if $d_v(v_i,w_s)$ satisfies the following criteria:

$$d_v(v_i,w_s) = \min_k d_v(v_i, w_k))$$

Where $d_v(v_i,w_k)$ is the distance between v_i and w_k. In our method, we want to find a set of cluster centers D_w so that $E(D_w)$ is minimized and D(S) is maximized. D(s) is the sum of distance between centers. $E(D_w)$ is calculated by:

$$E(D_w) = \sum_{w_s \in D_w} \sum_{v_i \in C(w_s)} d_v(v_i, w_s)$$

4.3. Creating the initial solution

Because we work with numerical variables, we use floating-point number vector. In our algorithm, one solution D_w contains coordinates of K cluster centers. D_w is a chromosome of genetic algorithm. Our population is a set of chromosomes ID_w

$$ID_w = \{D_{w,1}; D_{w,2}; D_{w,3} .. D_{w,n}\}$$

Where n is the population size(pop_size).

We build an initial solution set ID_w based on the set of potential cluster centers that were discovered by Kohonen's self-organizing algorithm. Figure 2 is a typical initial population. This population contains 3 chromosomes, each chromosome is the coordinates of 3 cluster centers.

6	6
9	8
13	12

	0
4	5

7	
8	
12	4

Figure 2: Initial population containing 3 chromosomes

4.4.Building the evaluation function

We calculate the fitness value of each chromosome by minimizing $E(D_w)$ and maximizing distance D(s).

Depending on the application, we choose two weights alpha and beta and transform two constraints into a problem of minimizing the following function:

$$E(D_{w,i}) = \text{alpha} * E(D_{w,i}) - \text{beta} * D(s)$$

The total fitness of population is calculated by

$$F = \sum_{i=1}^{pop\_size} E(D_{w,i})$$

The selection probability p_i for the chromosome $D_{w,i}$ is calculated by

$$Pi = E(D_{w,i}) / F$$

The cumulative probability q_i for the chromosome $D_{w,i}$ is calculated by:

$$qi = \sum_{j=1}^{i} pj$$

4.5. Defining the genetic operators (crossover and mutation)

4.5.1.Crossover operator

Based on the old population, we create new population. This session contains two steps:

Step1: Create the initial new population

a) Create a random (float) number r from the range [0,1]. If r<q1 then select the first chromosome else select the i^{th} chromosome so that r is in (q_{i-1}, qi).

b) Repeat this procedure until creating pop_size chromosomes for new population.

Step 2: For each chromosome in new population, generating a random (float) number r from the range [0,1], if $r<p_c$ (probability of crossover) then given chromosome will be selected for crossover. For each pair of chromosomes, we generate a random integer from [1..K]; and use this number for defining the position of crossover.

$D_{w,i} = (g_{w,i,1} \ldots g_{w,i,k} \ldots\ldots g_{wi,n})$

$D_{w,j} = (g_{w,j,1} \ldots g_{w,j,k} \ldots\ldots g_{wj,n})$

After crossover operator, we have two new chromosomes as follows:

$D_{w,i} = (g_{w,i,1} \ldots g_{w,j,k} \ldots\ldots g_{wi,n})$

$\Updownarrow$

$D_{w,j} = (g_{w,j,1} \ldots g_{w,i,k} \ldots\ldots g_{wj,n})$

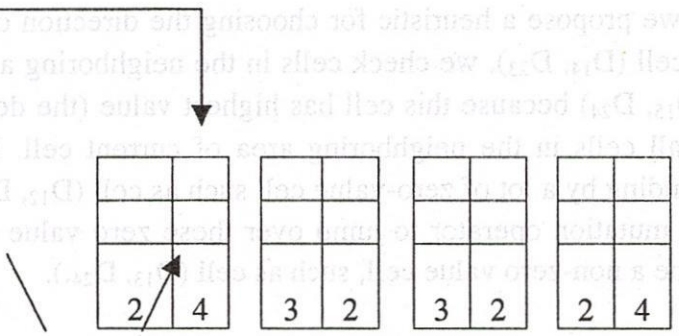

Figure 3: Crossover operator

4.5.2. Slight mutation operator

For each chromosome of current population, we generate a random (float) number r from the range [0,1], if $r<p_m$ (probability of mutation) then given chromosome will be selected for mutation. The position for mutation will be chosen by generating a random integer number from [1..K].

$D_{w,i} = (g_{w,i,1}, \ldots, g_{w,i,k}, \ldots, g_{wi,n})$

$\downarrow$

$D_{w,i} = (g_{w,i,1}, \ldots, X, \ldots, g_{wi,n})$

Where $X = g_{w,i,k} +/- \Delta x$ (Δx is an arbitrary constant)

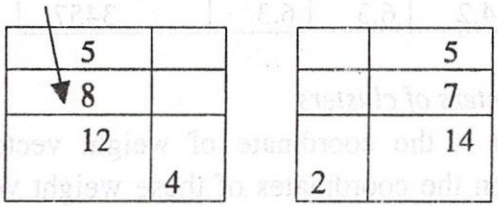

Figure 4: Mutation operator

4.5.3. Using MDDM for controlling the direction of mutation

Besides of crossover, the offspring can be built by mutation. After a number of generations, if the algorithm can not gather a better population, we use MDDM to improve the population. We use MDDM for choosing good direction of mutation. We call this kind of mutation is MDDM mutation. A MDDM with a lot of zero value cells is in table 3.

D25	60	0	11	21	0
D24	7	0	12	17	45
D23	0	0	0	8	0
D22	0	8	0	0	0
D21	0	0	0	6	8
	D11	D12	D13	D14	D15

Table 3: MDDM has a lot of zero value cells

Based on MDDM, we propose a heuristic for choosing the direction of mutation. If the current center is in cell (D_{14}, D_{23}), we check cells in the neighboring area of current cell and jump to cell (D_{15}, D_{24}) because this cell has highest value (the density of input vector is high) among all cells in the neighboring area of current cell. If our current center is in a cell surrounding by a lot of zero-value cell such as cell (D_{12}, D_{22}) in table 3. In this case, we control mutation operator to jump over these zero value cells, and the next potential cell may be a non-zero value cell, such as cell (D_{13}, D_{24}.).

5.EXPERIMENTAL RESULTS

We employed proposed method to discover clusters based on the marks in math, physics and chemistry of 5000 students. After transforming tuple of database into vector of MDDM and using Kohonen's self-organizing map algorithm, we have the data in table 4.

X1	X2	X3	# DAPTED TIMES
4.2	4.6	2.1	100090
2.1	0.4	0.6	8249
0.6	0.6	2.2	4656
8.3	6.3	4.2	3621
4.2	6.3	6.3	3457

Table 4: Potential centers of clusters

Where (X1, X2, X3) is the coordinate of weight vector. We build an initial population for GA based on the coordinates of these weight vectors. We run GA with pc=0.25 and pm=0.1, pop_size=50 and extract 45 generations of our testing and show them in table 5.

Gene#	InterDist	IntraDist	Mutation
1	21.2732	2.5827	Slight
2	21.2732	2.5827	Slight
3	21.2732	2.5827	Slight
4	21.2732	2.5827	Slight
5	21.2732	2.5827	Slight
6	22.2888	2.5290	Slight
2.5827 occurrence 4			
7	22.2888	2.5290	Slight
8	22.2888	2.5290	Slight
9	22.2888	2.5290	Slight
10	22.2888	2.5290	Slight
11	21.9938	2.5237	Slight
2.5290 occurrence 4			
12	22.9832	2.5201	Slight
13	22.9832	2.5201	Slight
14	22.9832	2.5201	Slight
15	22.9832	2.5201	Slight
16	22.9832	2.5201	Slight
17	24.2329	2.4003	Slight
2.5201 occurrence 4			
18	24.2329	2.4003	Slight
19	24.2329	2.4003	Slight
20	22.7780	2.3829	Slight
2.3829 occurrence 2			
21	22.7780	2.3829	Slight
22	22.7780	2.3829	Slight
2.3829 occurrence 2			
23	22.2355	2.3764	Slight
24	22.2355	2.3764	Slight
25	23.8846	2.2512	Slight
2.2512 occurrence 2			
26	23.8846	2.2512	Slight
27	23.8846	2.2512	Slight
28	23.8846	2.2512	Slight
29	23.8846	2.2512	Slight
30	23.8846	2.2512	Slight
31	23.8846	2.2512	Slight
MDDM mutation			
32	23.8846	2.2512	Slight
33	23.8846	2.2512	Slight

34	23.8846	2.2512	Slight
36	23.8846	2.2512	Slight
MDDM mutation			
37	23.8846	2.2512	Slight
38	23.8846	2.2512	Slight
39	23.8846	2.2512	Slight
40	23.8846	2.2512	Slight
41	23.8846	2.2512	Slight
MDDM mutation			
42	23.8846	2.2512	Slight
43	23.8846	2.2512	Slight
44	23.8846	2.2512	Slight
45	23.8846	2.2222	Slight
2.2512 occurrence		3	

Table5: Result of first 45 generations

In the above data, the IntraDist is decreased and InterDist is increased with time. From generation 27 to generation 31 the IntraDist and InterDist are not changed, we start MDDM mutation and in generation 45, these two factors are changed. After 300 generations, we have a solution that is shown in Table 6 where (X1, X2, X3) is the coordinate of cluster centers.

X1	X2	X3
1.000	1.000	1.000
3.500	3.500	2.500
9.000	9.000	7.000
6.000	6.000	4.500

Table 6: coordinate of centers of clusters

6.COMPARISON TO C-MEANS ALGORITHM AND OTHER RANDOM SEARCH BASED CLUSTERING ALGORITHM

In this section, we will compare our algorithm with some clustering algorithms such as c-means or random search based algorithm [7].

The c-means algorithm is a well-known algorithm for clustering process. This algorithm uses iterative approximation to search for the best partition of a given set of n objects into c clusters. The c-means algorithm can be summarized as follows:

1. Construct an arbitrary partition c x n matrix $U=(e^{t}ij)$ where e_{ij} is equal to one if object $X_{i(}x_{i1,} \ldots. x_{im})$ is in cluster C_j and to zero if not.

2.Calculate the center Vj of cluster Cj.

$V_{j\,(}v_{j1,..}\, v_{jk...,}v_{jm})$

3.Update the partition matrix U^{t+1} (e^{t+1}_{ij}) by calculating the distance d_{ij} between object Xi by and cluster Cj and modify the elements of partition matrix.

$e^{t+1}_{ij} = 1$ if $d_{ij} = \min\{d_{jk}\}$ for all k= 0..c and 0 if otherwise.

4.Determine the convergence of the matrix is achieved by comparing the difference with ε.

$$| U^{t+1} - U^{t} | < \varepsilon$$

With ε is a threshold of convergence.

If the partition matrix is not convergent go back to step 2 with the iteration number t=t+1, otherwise the clustering process is terminated.

If we use the c-means algorithm for this data set with 10 potential clusters, and each e_{ij} occupies 2 bytes then the partition matrix occupies 100,000 bytes (5,000x10x2). Because the c-means algorithm uses two partition matrices, one for time t and another for time t+1, the total of bytes needed for two matrices is 200,000 bytes. With our algorithm, we use 5 intervals for each dimension, the number of bytes needed for MDDM is 250 bytes (5x5x5x2). For GA, we created 50 chromosomes, each chromosome has 10 potential cluster centers and each center occupies 24 bytes (8x3). The number of needed bytes is 12.000 bytes.

This data set contains only 5000 objects, in the typical data mining application, the number of objects may exceed 100,000 or one million and in this case, c-means algorithm can not be implemented efficiently by loading the whole matrix into the main memory.

We also compare our method with the random based clustering algorithm to be proposed in [7]. The search space of this algorithm is a graph, each node of this graph is a set of k cluster centers and two neighboring nodes are connected by an edge. Neighboring nodes are nodes that differ in only one cluster center. This algorithm will search node in state space that satisfies a given fitness function. Firstly, an arbitrary node in search space will be chosen for potential nodes (solutions) then the algorithm will choose randomly a node in the neighboring area of potential node. If this node is better than potential node, it will become the potential node. The process continues until a given fitness value is reached.

We employed MDDM and Kohonen's self-organizing algorithm to discover the number of potential cluster centers. This number can help to build nodes in the search space needed for this algorithm. For large data set, we utilized the strength of GA that can search concurrently in many nodes on search space instead of searching in only one node in algorithm to be proposed in [7]. The analysis of GA searching can be referred in [11]. Besides of crossover, our algorithm still employs the MDDM mutation to increase the speed of convergence.

7.APPLICATION TO HOUSE BUYING AND SELLING SERVICE

We applied this method to discover knowledge from house buying and selling service in HoChiMinh City, Vietnam. The purpose of our application is to extract knowledge from information to be collected from public media and the experience of experts working in this field. Some variables that are used for clustering process are listed below:

House price, length of house, width of house, area of land, area of house, distance from the center of HoChiMinh city, house location, house level, number of floors, number of toilets...

We proposed a set of weights for scaling the importance of variables in our clustering process and use Euclidean distance as a measure of similarity. After using the proposed algorithm, we created rules based on the discovered clusters. Some rules are appropriated to the thinking of the experts such as following rules:

- If area of house < 60 squared meters and house locates in narrow then the house price < 30,000 USD.
- If area of house < 60 squared meters and house locates near a market then the house price > 40,000USD.

Some rules are very interesting and they can help us to analyze the trend in house market, although the number of objects that satisfied these rules is small.

8.CONCLUSIONS

We gathered some preliminary results in using MDDM, Kohonen's self-organizing algorithm and GA to discover the clusters in large database with the numerical attributes. The data set is very large, the heuristic and measure for improving the speed of mining is necessary to be proposed. The experiments show very encouraging results in using this combined model for clustering discovery. We continue to study how to use this combined model in discovery clusters from large databases that concurrently contains numerical, nominal, conceptual hierarchy fields and to use fuzzy logic membership function in building the fitness function.

KẾT HỢP MÔ HÌNH DỮ LIỆU ĐA CHIỀU, THUẬT TOÁN TỰ TỔ CHỨC VÀ THUẬT TOÁN DI TRUYỀN ĐỂ KHÁM PHÁ CLUSTER TRONG KHAI MỎ DỮ LIỆU

Hoang Kiem – Do Phuc

TÓM TẮT: Khai mỏ dữ liệu là khám phá các mẫu tiềm ẩn trong các cơ sở dữ liệu lớn. Trong bài báo này, chúng tôi nghiên cứu một mô hình kết hợp để khám phá cluster. Chúng tôi xây dựng mô hình dữ liệu đa chiều (MDDM) , kế đến chúng tôi dùng thuật toán tự tổ chức để phát hiện các cluster tiềm năng và thuật toán di truyền(GA) để kiểm chứng các cluster này.

Bằng cách kết hợp MDDM với GA, chúng tôi đề xuất một heuristic để nâng cao hiệu suất của bài toán khám phá cluster.

TÀI LIỆU THAM KHẢO

[1] L.P.J Veelenturf, Analysis and application of Artificial Neural Networks, Prentice hall, 1995,183-191

[2] Hair Anderson, Multivariate Data Analysis, MacMillan Publishing Company, 1990, 340-435

[3] Hoang Kiem, Do Phuc, Using MDDM for mining association rules in a large database, proceedings of IT@EDU98 conference, VNU-HCMC, Vietnam, 1998, 6.6.1-6.6.8

[4] Jun Yan, Using Fuzzy logic, Prentice Hall, 1994,72-76

[5]O.Strout, Chemical Pattern Recognition, Research studies Press, England, 1986, 71-78

[6]Pieter Adrians, Dolf Zantige, Data Mining, Addision Wesley, Longman, 1996, 72-76

[7] Raymond T Ng, Jiawei Han, Efficient and Effective clustering methods for spatial data mining, Proceedings of the 20th VLDB conference, Santiago, Chile, 1994

[8] Timothy J. Ross, Fuzzy logic with engineer application, Mac Graw Hill, 1995, 274-282

[9] Zbigniew Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs, Springer-Verlag, 1992, 13-42

[10] Bezdek, J. C. Cluster validity with fuzzy sets. In J. Cybernetics, Vol. 3, No.3, 1974, 58-73

[11] David E. Goldberg. Genetic Algorithm in Search, Optimization, and Machine learning, Addison-Wesley Publishing Company, Inc, 1989,147-2