TELKOMNIKA, Vol. 11, No. 4, April 2013, pp. 2050~2057
ISSN: 2302-4046

The K-Medoids Clustering Algorithm with Membrane Computing

Yuzhen Zhao*[1], Xiyu Liu[2], Hua Zhang[3]
[1,2]School of Management Science and Engineering, Shandong Normal University, Jinan, China
[3]Yellow River Road Bridge Engineering Company, Jinan, China
*Corresponding author, e-mail: zhaoyuzhen_happy@126.com

Abstract

The K-medoids clustering algorithm is realized by a P system in this paper. Because the membrane system has great parallelism and lower computational time complexity, it is suitable for solving combinatorial problems like the clustering problem. A P system with all the rules to solve the K-medoids algorithm was constructed. The specific P system is associated with the dissimilarity matrix between n objects. This system can get one possible classifications in a non-deterministic way. Through example test, it is appropriate for cluster analysis. This is a new attempt in applications of membrane system and it provides new ideas and methods for cluster analysis.

***Keywords**: Clustering algorithm, the K-medoids clustering, Membrane computing, P System*

1. Introduction

Clustering is a very important problem in machine learning, statistics, biology, data mining and many other many fields. Through the process of clustering, data set are partitioned into clusters with intra-cluster data similar and inter-cluster data dissimilar. From another perspective, the clustering problem is actually a combination problem of data, which is to find a program that meets the above conditions from all combinations [1].

Many fields like combinatorial problem, finite state problems and graph theory have applied membrane computing. For many combinatorial problems membrane computing approaches are very suitable used on account of the vast parallelism. The time complexity parallelism of the computing will be lessened so it can meet the requirement of improving the processing speed of the big data [2, 3].

The k-medoids algorithm is one of the partitioning algorithms for spatial data. It is the combinatorial problem so it can be solved with membrane computing [4].

This paper combines these two above to solve the problem of clustering n objects to k clusters. It uses the subscript i of point a_i to represent the i-th object of the original objects and it uses the dissimilarity of any two original objects to define the distance of corresponding points a. Different clusters are represented by different membranes. First, it randomly assigns center points of the k clusters, and the remaining points entry into membranes with the nearest center point. Second, it re-specifies the center point of each membrane making the summation of the dissimilarity between the center point and all other points in the membrane the shortest. And then it redistributes the remaining points, and so on, until all the center points no longer change. In this way, it gains the clusters. Finally, it puts the information out in the form of character strings into the output membrane. This strategy is a new application of membrane computing.

2. The K-Medoids Algorithm

The K-medoids algorithm which is more robust to outliers and noise is a one of the classical partitioning algorithm of clustering improved by the K-means algorithm [5].

The data set $X = \{x_1, x_2, ..., x_n\}$ with n objects can be clusters into k clusters by it. A medoid is one object of a cluster with the minimal total distance to all other objects. By distributing all non-medoid objects to the nearest medoid the clusters $C = \{C_1, C_2, ..., C_k\}$ are generated. They have the following three properties:

Received January 6, 2013; Revised February 22, 2013; Accepted March 4, 2013

1) $C_i \neq \varnothing \ 1 \le i \le k$;
2) $\bigcup_{i=1}^{k} C_i = X$;
3) $C_i \bigcap C_j = \varnothing, i \neq j, 1 \le i, j \le k$.

What's more, objects in the same cluster are similar to each other and objects from distinct clusters are different from each other. The distance needs to be defined in order to find the solution. In the literature various alternatives have been reported to approach this task. It can choose one according to the request.

This paper uses the dissimilarity to define it. First of all, it defines the dissimilarity matrix D_{nn}' between any two objects as follows:

$$D_{nn}' = \begin{pmatrix} w_{11}' \ w_{12}' .. & w_{1n}' \\ w_{21}' \ w_{22}' .. & w_{2n}' \\ ... & \\ w_{n1}' \ w_{n2}' .. & w_{nn}' \end{pmatrix} \quad (1)$$

Where, w_{ij}' is the dissimilarity between the i-th object and the j-th object [6] . Specific calculation method is selected depending on the type of object.

Then, it changes the matrix elements w_{ij}' into integer w_{ij} by rounding for membrane computing. By this, it gains the new matrix D_{nn} as follows [7]:

$$D_{nn} = \begin{pmatrix} w_{11} \ w_{12} .. & w_{1n} \\ w_{21} \ w_{22} .. & w_{2n} \\ ... & \\ w_{n1} \ w_{n2} .. & w_{nn} \end{pmatrix} \quad (2)$$

The K-medoid algorithm has some specific methods. Partitioning around medoids (PAM) is a representative one.

The steps of the PAM are as follows:
1) Select k objects as medoids arbitrarily from all the objects as the initial k clusters.
2) Distribute the remaining objects to their most similar cluster with the shortest distance.
3) Randomly select non-medoid object O' .
4) Compute the distance of O' and all the other objects.
5) Set O' as the new medoid if the total distance is decreased.
6) Repeat the steps 2 to 5 above until all medoids don't change anymore [8].

So given arbitrary n objects, it can cluster them into k clusters by computing their dissimilarity matrix D_{nn} and using the K-mediods algorithm to cluster.

3. The K-Medoids Clustering Algorithm with Membrane Computing

3.1. The P System for the K-medoids Clustering Method

In this section, a P system for the K-mediods algorithm is proposed. Its structure is depicted in Figure 1.

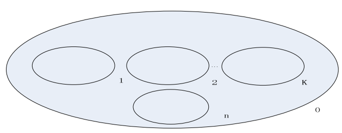

Figure 1. The P System for the K-Medoids Clustering Method

It use the subscript i of the points a_i to represent the i-th object of the original objects and use the matrix D_{nn} to compare the similarity between the n objects. The specific algorithm is followed:

First, it set the maximum data in the matrix D_{nn} Max, the minimum data in the matrix D_{nn} Min and set the absolute value of these two Abs for convenience.

The P system for clustering is defined as follows:

$$\Pi = (O, \mu, M_0, M_1, ..., M_k, M_n, R_0, R_1, ..., R_k, R_n, \rho) \quad (3)$$

Where:

1) $O=\{a_1, a_2, ..., a_n, \delta_1, s_0, \beta_1, e\}$. O represents the collection of objects in the P system.
2) $\mu=[_0[_1]_1[_2]_2 ..[_k]_k[_n]_n]_0$. μ represents the membrane structure of the P system.
3) $M_0 = \{\delta_1^k, \beta_1, a_1, a_2, ..., a_n, e\}, M_1 = M_2 = ... = M_k = \{s_0, \delta_1\}, M_n = \lambda$. M represents the collection of initial objects in each membrane.is the output membrane of this P system.

The rules in R_0 [9,10]:

$r_1 = \{a_i \beta_t \rightarrow A_{ti} A_{ti\ in_t} \beta_{t+1}\ 1 \le i \le n, 1 \le t \le k\}$

$r_2 = \{\delta_i^k a_i A_{1j_1} A_{2j_2} ... A_{kj_k} \rightarrow a_i A_{1j_1} A_{2j_2} ... A_{kj_k} U_{i1}^{w_{ij_1}} U_{i2}^{w_{ij_2}} ... U_{ik}^{w_{ij_k}} \mid 1 \le i, j_1, j_2, ..., j_k \le n\}$

$\cup \{(\delta_i^k \rightarrow \delta_{i+1}^k)_{\neg a_i} \mid 1 \le i \le n-1\}$

$\cup \{\delta_n^k a_n A_{1j_1} A_{2j_2} ... A_{kj_k} \rightarrow a_n U_{n1}^{w_{nj_1}} U_{n2}^{w_{nj_2}} ... U_{nk}^{w_{nj_k}} \mid 1 \le j_1, j_2, ..., j_k \le n\}$

$\cup \{(\delta_n^k A_{1j_1} A_{2j_2} ... A_{kj_k} \rightarrow \lambda)_{\neg a_n} \mid 1 \le j_1, j_2, ..., j_k \le n\}$

$r_3 = \{a_i U_{i1}^{t_1} U_{i2}^{t_2} ... U_{ik}^{t_k} \rightarrow a_{i\ in_j} \delta_{i+1}^k \mid t_j = 0, 1 \le i < n, 1 \le j \le k\}$

$\cup \{a_n U_{n1}^{t_1} U_{n2}^{t_2} ... U_{nk}^{t_k} \rightarrow a_{i\ in_j} \mid t_j = 0, 1 \le j \le k\}$

$r_4 = \{U_{i1}^{j_1} U_{i2}^{j_2} ... U_{ik}^{j_k} \rightarrow U_{i1}^{j_1-1} U_{i2}^{j_2-1} ... U_{ik}^{j_k-1} \mid 1 \le i \le n, 1 \le j_s \le Max, 1 \le s \le k\}$

$r_5 = \{(e^i \psi^j \rightarrow (\theta)_{in_1} (\theta)_{in_2} ... (\theta)_{in_k} \mid 1 \le i \le n, 0 \le \psi < k)\}$

$\cup \{(\psi^k \rightarrow (\alpha)_{in_1} (\alpha)_{in_2} ... (\alpha)_{in_k})\}$

$r_6 = \{\varphi \alpha w \rightarrow (\alpha w)_{in_n}\ w \in A_{jt} \cup \{a_p \mid 1 \le p \le n\}^*\}$

The rules in $R_i (1 \le i \le k)$:

$r_i' = \{(\alpha w \rightarrow \alpha w a_i)_{a_i} \mid 1 \le i \le n, w \in A_{jt} \cup \{a_p \mid 1 \le p \le n\}^*\}$

$r_{n+1}' = \{\alpha w \rightarrow (\varphi \alpha w, out)\ w \in A_{jt} \cup \{a_p \mid 1 \le p \le n\}^*\}$

$r_{n+2}' = \{(\theta A_{jh})_{\neg a_i} \rightarrow A_{jh} (\psi, A_{jh}, \delta_1, out) \# \mid 1 \le i, h \le n, 1 \le j \le k\}$

$r_{n+3}' = \{\theta \delta_i a_i \rightarrow \theta \zeta_i O_i \mid 1 \le i \le n\} \cup \{(\theta \delta_i \rightarrow \theta \delta_{i+1})_{\neg a_i} \mid 1 \le i \le n\}$

$r_{n+4}' = \{s_t O_i a_j A_{hp} \rightarrow b_j O_i s_{t+w_{ij}-w_{pj}} A_{hp} \mid 1 \le i, j, p \le n, 1 \le h \le k, |t| \le nAbs\}$

$r_{n+5}' = \{s_i A_{jp} O_h \rightarrow s_0 A_{jh} a_p \eta \mid -nAbs \le i < 0, 1 \le j \le k, 1 \le p, h \le n\}$

$\cup \{s_i A_{jp} O_h \rightarrow s_0 A_{jp} a_h \sigma\} \mid 0 \le i \le nAbs, 1 \le j \le k, 1 \le p, h \le n\}$

$r_{n+6}' = \{b_i \rightarrow a_i \mid 1 \le i \le n\}$

$r_{n+7}' = \{\zeta_i \rightarrow \delta_{i+1} \mid 1 \le i \le n\}$

$r_{n+8}' = \{\eta^i \sigma^j \rightarrow (e, out) \mid 1 \le i \le n, 0 \le j \le n\} \cup \{(\sigma^i \rightarrow (\psi, out))_{\neg \eta^j} \mid 1 \le i, j \le n\}$

$r_{n+9}' = \{a_i \rightarrow (a_i, out) \mid 1 \le i \le n\}$

$r_{n+10}' = \{\delta_{n+1} A_{jp} \theta \rightarrow (\delta_1 A_{jp}, out) \delta_1 A_{jp} \# \mid 1 \le j \le k, 1 \le p \le n\}$

$\cup \{(A_{jp} \theta)_{\neg \delta_{n+1}} \rightarrow (\delta_1 A_{jp}, out) \delta_1 A_{jp} \# \mid 1 \le j \le k, 1 \le p \le n\}$

$\rho = \{r_i > r_j \mid 1 \le i < j \le 6\} \cup \{r_i' > r_j' \mid 1 \le i < j \le n+10\}$

3.2. An Overview of Computations

The rule r_1 is executed firstly according to the priority relationship. It chooses one from all the points a_i arbitrarily and put the point into membrane 1 which represents the first cluster. At the same time, it converts the chosen lowercase letter a_i into corresponding capital letter A_{ti} to avoid putting it into the other membranes and to distinguishing the center point and other common data point. The subscripts of A_{ti} show that the center point of cluster t is the i-th object. Object β_t is used to control the time of execution of rules and the serial number of membrane which it put the center point into. According role r_1, it successfully choose k center points arbitrarily from all the numbers and put them into membranes labeled from 1 to k which represent the k clusters.

Then it executes the rule r_2 .Beginning with point a_1 , the objects U_{ij}^t are generated. The corner marks i, j and t show that the value of dissimilarity between point a_i and the center point of cluster j is t. Then let all the superscript decrease at the same time until one of them is 0.This mean point a_1 is the most similar to center point of the corresponding cluster. Next it puts point a_1 into the corresponding membrane while δ_1 increases, making it begin to calculate the dissimilarity between point a_2 and all center points. If the point a_i does not exist, meaning that this point has been chosen to be a center point, then δ_i add 1 directly and goes into the next cycle. And so on until a_n . If a_n exists, it generates the dissimilarity between it and each center point. Then let the A objects in 0 membrane disappear and the superscript of objects U_{ij}^t decrease at the same time until one of them is 0. Next it put point a_n into the corresponding membrane. Objects U_{ij}^t disappear. If point a_n has been selected as a center point, it cannot produce the objects U_{ij}^t and the auxiliary point δ_n and A disappear. All points enter into the corresponding membrane at this time. Clusters have been completed the first step: it gives k center points arbitrarily, and divides the remaining points into k clusters based on the dissimilarity between them and the center points. At this time, there is no point a_i in membrane 0. Because there is an object e in membrane 0, it generates object θ into membranes labeled from 1 to k by rule r_5 . Then rules in membranes labeled from 1 to k are triggered. Rule r_5 makes the redetermination of center points goes after all the points a_i are divided into k clusters. If there are k objects ψ in membrane 0, this mean that the center points of the k membranes labeled from 1 to k are not changed.

Then, it executes rules in membranes labeled from 1 to k. If there is no point except the center point in one membrane, it uses the object # to stop the process and put an object ψ to show it. Otherwise it begins to find the best center points. It starts with a_1 . If the point does not exist, δ_1 adds 1 and goes into the next cycle directly. If the point exists, it is set as the new center point. Then calculate the difference between the dissimilarity between the new center point and the remaining point and the dissimilarity between the original center point and the remaining point and add the data to the subscript of object S. The lowercase a_j is replaces by b_j after the calculation to avoid repeating calculation. If the subscript of S is less than 0 after calculating the dissimilarity with all remaining points, the point a_1 can reduce the total consumption. So it is set as the new center point and produce an object η to inform the new center point has been generated. If the subscript of S is greater than or equal to 0, the new center point can't reduce the total consumption. Therefore it maintains the original center point, rename object O_h to a_h and produce an object σ to show it. The object δ_{i+1} is produced to go to the next cycle to compare the object a_2 and the center point and so on until all the objects a_i are compared. If there is any η in this membrane, it put an object e out to inform that the data need to be reclassified out of the membrane. And put an object ψ out to inform that the center point this is membrane is not changed. Then a_i are put out . Last it restores objects in this membrane to initial state and put the information of the center point out. It uses the object $\#$ to stop the computation step sending an object δ_1 out to show it.

When responses in membranes labeled from 1 to k are completed, the membrane 0 includes triggering rule and then the process of clustering are repeated until no e in membrane 0 showing that center points in all k membranes don't change and the clustering adjustment process end. Then membranes labeled from 1 to k accept an object α to execute the rules from r_1' to r_{n+1}' to output the result of the cluster. The result is in the form of string. Finally, put these strings formed above to the output membrane n. One computation process is over.

3.3. Test and Analysis

To illustrate how the membrane system shown in Fig.1 run specifically, the following simple example is considered: cluster 7 integral points (1,3), (1,4), (2,1), (2,4), (3,2), (4,1), (4,3) shown in Figure 2 into two classes. When changing the center point of each cluster. Obviously, n=7, k=2.

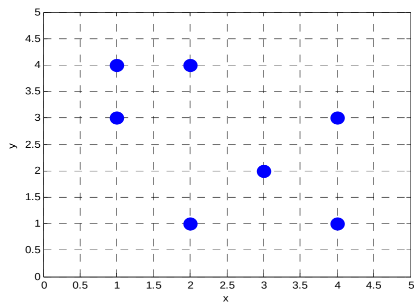

Figure 2. The 7 Points Waiting for being Clustered

First of all, it defines the dissimilarity matrix D_{77}'. In this example, it uses the square of distance between any two points as the dissimilarity. Because the points are integral points, matrix D_{77} is the same to matrix D_{77}':

$$D_{77} = D_{77}' = \begin{pmatrix} 0 & 1 & 5 & 2 & 5 & 13 & 9 \\ 1 & 0 & 10 & 1 & 8 & 18 & 10 \\ 5 & 10 & 0 & 9 & 2 & 4 & 8 \\ 2 & 1 & 9 & 0 & 5 & 13 & 5 \\ 5 & 8 & 2 & 5 & 0 & 2 & 2 \\ 13 & 18 & 4 & 13 & 2 & 0 & 4 \\ 9 & 10 & 8 & 5 & 2 & 4 & 0 \end{pmatrix} \quad (4)$$

The membrane system clustering these seven numbers into two classes is shown in Figure 3:

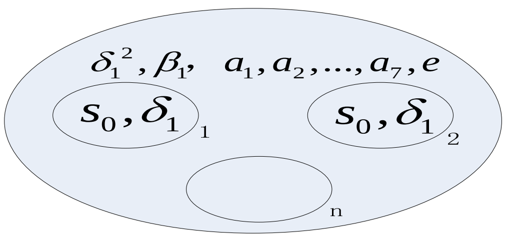

Figure 3. The P System Clustering Seven Numbers into Two Clusters

Steps of one circulation of the clustering are listed in Table 1. Because the steps are almost the same, only parts of them are listed here.

Table 1. Steps of the First Circulation of the Clustering

t	*membrane 0*	*membrane 1*	*membrane 2*
0	$\delta_1^2, \beta_1, a_1, a_2, \ldots, a_7, e$	s_0, δ_1	s_0, δ_1
1	$\delta_1^2, \beta_2, A_{11}, a_2, \ldots, a_7, e\ (r_1)$	s_0, δ_1, A_{11}	s_0, δ_1
2	$\delta_1^2, \beta_3, A_{11}, A_{22}, a_3, \ldots, a_7, e\ (r_1)$	s_0, δ_1, A_{11}	s_0, δ_1, A_{22}
3	$\delta_2^2, \beta_3, A_{11}, A_{22}, a_3, \ldots, a_7, e\ (r_2)$	s_0, δ_1, A_{11}	s_0, δ_1, A_{22}
4	$\delta_3^2, \beta_3, A_{11}, A_{22}, a_3, \ldots, a_7, e\ (r_2)$	s_0, δ_1, A_{11}	s_0, δ_1, A_{22}
5	$\beta_3, A_{11}, A_{22}, a_3, \ldots, a_7, e, U_{31}^5, U_{32}^{10}\ (r_2)$	s_0, δ_1, A_{11}	s_0, δ_1, A_{22}
6	$\beta_3, A_{11}, A_{22}, a_3, \ldots, a_7, e, U_{31}^4, U_{32}^9\ (r_4)$	s_0, δ_1, A_{11}	s_0, δ_1, A_{22}
…			
10	$\beta_3, A_{11}, A_{22}, a_3, \ldots, a_7, e, U_{31}^0, U_{32}^5\ (r_4)$	s_0, δ_1, A_{11}	s_0, δ_1, A_{22}
11	$\delta_4^2, \beta_3, A_{11}, A_{22}, a_4, \ldots, a_7, e\ (r_3)$	$s_0, \delta_1, A_{11}, a_3$	s_0, δ_1, A_{22}
…			
47	$\beta_3, e\ (r_3)$	$s_0, \delta_1, A_{11}, a_3, a_5, a_6, a_7$	$s_0, \delta_1, A_{22}, a_4$
48	$\beta_3\ (r_5)$	$s_0, \delta_1, A_{11}, a_3, a_5, a_6, a_7, \theta$	$s_0, \delta_1, A_{22}, a_4, \theta$
49	β_3	$s_0, \delta_2, A_{11}, a_3, a_5, a_6, a_7, \theta (r_{10}')$	$s_0, \delta_2, A_{22}, a_4, \theta (r_{10}')$
50	β_3	$s_0, \delta_3, A_{11}, a_3, a_5, a_6, a_7, \theta (r_{10}')$	$s_0, \delta_2, A_{22}, a_4, \theta (r_{10}')$
51	β_3	$s_0, A_{11}, O_3, a_5, a_6, a_7, \theta, \zeta_3 (r_{10}')$	$s_0, \delta_4, A_{22}, a_4, \theta (r_{10}')$
52	β_3	$s_{-3}, A_{11}, O_3, b_5, a_6, a_7, \theta, \zeta_3 (r_{11}')$	$s_0, A_{22}, O_4, \theta (r_{10}')$
53	β_3	$s_{-12}, A_{11}, O_3, b_5, b_6, a_7, \theta, \zeta_3 (r_{11}')$	$s_0, A_{22}, a_4, \theta, \sigma (r_{12}')$
54	β_3, ψ	$s_{-13}, A_{11}, O_3, b_5, b_6, b_7, \theta, \zeta_3 (r_{11}')$	$s_0, A_{22}, a_4, \theta (r_{15}')$
55	β_3, ψ, a_4	$s_0, A_{13}, a_1, b_5, b_6, b_7, \theta, \eta, \zeta_3 (r_{12}')$	$s_0, A_{22}, \theta (r_{16}')$
56	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_0, A_{13}, a_1, a_5, b_6, b_7, \theta, \eta, \zeta_3 (r_{13}')$	$s_0, A_{22}, \delta_1 (r_{17}')$
57	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_0, A_{13}, a_1, a_5, a_6, b_7, \theta, \eta, \zeta_3 (r_{13}')$	s_0, A_{22}, δ_1
t	*membrane 0*	*membrane 1*	*membrane 2*
58	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_0, A_{13}, a_1, a_5, a_6, a_7, \theta, \eta, \zeta_3 (r_{13}')$	s_0, A_{22}, δ_1
59	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_0, A_{13}, a_1, a_5, a_6, a_7, \theta, \eta, \delta_4 (r_{14}')$	s_0, A_{22}, δ_1
60	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_0, A_{13}, a_1, a_5, a_6, a_7, \theta, \eta, \delta_5 (r_{10}')$	s_0, A_{22}, δ_1
61	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_0, A_{13}, a_1, O_5, a_6, a_7, \theta, \eta, \zeta_5 (r_{10}')$	s_0, A_{22}, δ_1
62	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_0, A_{13}, b_1, O_5, a_6, a_7, \theta, \eta, \zeta_5 (r_{11}')$	s_0, A_{22}, δ_1
63	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_{-2}, A_{13}, b_1, O_5, b_6, a_7, \theta, \eta, \zeta_5 (r_{11}')$	s_0, A_{22}, δ_1
64	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_{-8}, A_{13}, b_1, O_5, b_6, b_7, \theta, \eta, \zeta_5 (r_{11}')$	s_0, A_{22}, δ_1
65	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_0, A_{15}, b_1, a_3, b_6, b_7, \theta, \eta^2, \zeta_5 (r_{12}')$	s_0, A_{22}, δ_1
66	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_0, A_{15}, a_1, a_3, b_6, b_7, \theta, \eta^2, \zeta_5 (r_{13}')$	s_0, A_{22}, δ_1
67	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_0, A_{15}, a_1, a_3, a_6, b_7, \theta, \eta^2, \zeta_5 (r_{13}')$	s_0, A_{22}, δ_1
68	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_0, A_{15}, a_1, a_3, a_6, a_7, \theta, \eta^2, \zeta_5 (r_{13}')$	s_0, A_{22}, δ_1
69	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_0, A_{15}, a_1, a_3, a_6, a_7, \theta, \eta^2, \delta_6 (r_{14}')$	s, A_{22}, δ_1
70	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_0, A_{15}, a_1, a_3, O_6, a_7, \theta, \eta^2, \zeta_6 (r_{10}')$	s_0, A_{22}, δ_1
71	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_8, A_{15}, b_1, a_3, O_6, a_7, \theta, \eta^2, \zeta_6 (r_{11}')$	s_0, A_{22}, δ_1
72	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_{10}, A_{15}, b_1, b_3, O_6, a_7, \theta, \eta^2, \zeta_6 (r_{11}')$	s_0, A_{22}, δ_1
73	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_{12}, A_{15}, b_1, b_3, O_6, b_7, \theta, \eta^2, \zeta_6 (r_{11}')$	s_0, A_{22}, δ_1
74	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_0, A_{15}, b_1, b_3, a_6, b_7, \theta, \eta^2, \zeta_6, \sigma (r_{12}')$	s_0, A_{22}, δ_1
75	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_0, A_{15}, a_1, b_3, a_6, b_7, \theta, \eta^2, \zeta_6, \sigma (r_{13}')$	s_0, A_{22}, δ_1
76	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_0, A_{15}, a_1, a_3, a_6, b_7, \theta, \eta^2, \zeta_6, \sigma (r_{13}')$	s_0, A_{22}, δ_1
77	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_0, A_{15}, a_1, a_3, a_6, a_7, \theta, \eta^2, \zeta_6, \sigma (r_{13}')$	s_0, A_{22}, δ_1
78	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_0, A_{15}, a_1, a_3, a_6, a_7, \theta, \eta^2, \delta_7, \sigma (r_{14}')$	s_0, A_{22}, δ_1
79	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_0, A_{15}, a_1, a_3, a_6, O_7, \theta, \eta^2, \zeta_7, \sigma (r_{10}')$	s_0, A_{22}, δ_1
80	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_4, A_{15}, b_1, a_3, a_6, O_7, \theta, \eta^2, \zeta_7, \sigma (r_{11}')$	s_0, A_{22}, δ_1
81	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_6, A_{15}, b_1, b_3, a_6, O_7, \theta, \eta^2, \zeta_7, \sigma (r_{11}')$	s_0, A_{22}, δ_1
82	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_8, A_{15}, b_1, b_3, b_6, O_7, \theta, \eta^2, \zeta_7, \sigma (r_{11}')$	s_0, A_{22}, δ_1
83	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_0, A_{15}, b_1, b_3, b_6, a_7, \theta, \eta^2, \zeta_7, \sigma (r_{12}')$	s_0, A_{22}, δ_1
84	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_0, A_{15}, a_1, b_3, b_6, a_7, \theta, \eta^2, \zeta_7, \sigma (r_{13}')$	s_0, A_{22}, δ_1
85	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_0, A_{15}, a_1, a_3, b_6, a_7, \theta, \eta^2, \zeta_7, \sigma (r_{13}')$	s_0, A_{22}, δ_1
86	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_0, A_{15}, a_1, a_3, a_6, a_7, \theta, \eta^2, \zeta_7, \sigma (r_{13}')$	s_0, A_{22}, δ_1
87	$\beta_3, \psi, a_4, \delta_1, A_{22}$	$s_0, A_{15}, a_1, a_3, a_6, a_7, \theta, \eta^2, \delta_8, \sigma (r_{14}')$	s_0, A_{22}, δ_1
88	$\beta_3, \psi, a_4, e, \delta_1, A_{22}$	$s_0, A_{15}, a_1, a_3, a_6, a_7, \theta, \delta_8 (r_{15}')$	s_0, A_{22}, δ_1
89	$\beta_3, \psi, a_1, a_4, e, \delta_1, A_{22}$	$s_0, A_{15}, a_3, a_6, a_7, \theta, \delta_8 (r_{16}')$	s_0, A_{22}, δ_1
90	$\beta_3, \psi, e, a_1, a_3, a_4, \delta_1, A_{22}$	$s_0, A_{15}, a_6, a_7, \theta, \delta_8 (r_{16}')$	s_0, A_{22}, δ_1
91	$\beta_3, \psi, e, a_1, a_3, a_4, a_6, \delta_1, A_{22}$	$s_0, A_{15}, a_7, \theta, \delta_8 (r_{16}')$	s_0, A_{22}, δ_1
92	$\beta_3, \psi, e, a_1, a_3, a_4, a_6, a_7, \delta_1, A_{22}$	$s_0, A_{15}, \theta, \delta_8 (r_{16}')$	s_0, A_{22}, δ_1
93	$\beta_3, \psi, e, a_1, a_3, a_4, a_6, a_7, A_{15}, \delta_1^2, A_{22}$	$s_0, A_{15}, \delta_1 (r_{17}')$	s_0, A_{22}, δ_1

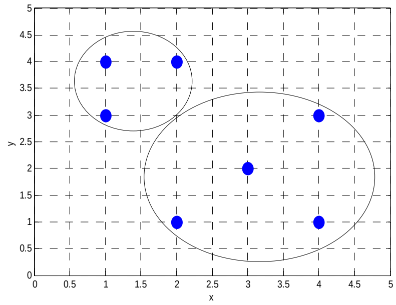

Figure 4. The Result of the Cluster of these 7 Points

In the end, these 7 points are clustered into two classes shown in Figure 4. The points in the same classes are closer to each other. So the right result of this method of clustering is gained.

4. Conclusion

This paper constructs a P system to realize the K-medoids clustering algorithm. This algorithm is suitable for cluster analysis by example test, but it needs to be further studied whether it is suitable for cluster analysis of large amount of data. Speaking from a theoretical point of view, the P system has great parallelism. So it can reduce the time complexity of computing and increases the computational efficiency. The following research work will focus on the theoretically analyze of the algorithm's time complexity. Additionally, membrane computing is a new biological computing method. Now its theoretical research is mature, but its application is not particularly extensive. A lot of applications will emerge in various fields in the future. The application in cluster proposed in this paper is one example. There are many clustering method and this paper only use the k-medoids algorithm. Membrane computing can be applied to a variety of other clustering methods.

Acknowledgments

The authors thank the reviewers for their suggestions. This project was supported by the Natural Science Foundation of China(No.61170038), Natural Science Foundation of Shandong Province, China (No.ZR2011FM001), Humanities and Social Sciences Project of Ministry of Education, China (No.12YJA630152), Social Science Fund of Shandong Province, China (No.11CGLJ22), Science-Technology Program of the Higher Education Institutions of Shandong Province, China (No.J12LN22), Science-Technology Program of the Higher Education Institutions of Shandong Province, China (No.J12LN65), Science-Technology Program of the Higher Education Institutions of Shandong Province, China (No.J12LN22), and Research Award Foundation for Outstanding Young Scientists of Shandong Province, China (No.BS2012DX041).

References

[1] Zhang HY. The research on clustering algorithm based on DNA computing. *PhD Thesis*. Jinan. Dept. Management Science and Engineering, Shandong Normal Univ., 2011.
[2] Chen H Z. Application Research on Membrane Computing. *PhD Thesis*. Chongqing. Dept. Computer Science, Chongqing Univ., 2011.
[3] Liang H. Research on Membrane Computing Optimization Methods. *PhD Thesis*. Hangzhou. Dept. Control Science and Engineering, Zhejiang Univ., 2007.
[4] Chandrashekhar AM, Raghuveer K. Performance evaluation of data clustering techniques using KDD Cup-99 Intrusion detection data set. *International Journal of Information and Network Security.* 2012; 1(4): 294–305.

[5] Remani NVJM, Kurra RR. Efficient K-Means Fuzzy Cluster Reliability on Angle Oriented Face Recognition. *International Journal of Informatics and Communication Technology.* 2013; 2(1): 180-187.
[6] Cardona M, Colomer MA, Pérez-Jiménez MJ. Hierarchical clustering with membrane computing. *Computing and informatics.* 2008; Feb(3): 497–513.
[7] Zhang HY, Liu XY. A CLIQUE algorithm using DNA computing techniques based on closed-circle DNA sequences. *Biosystems*. 2011; 105(1): 73–82.
[8] Han J, Kambr M. *Data Mining Concepts and Techniques*. 2nd ed. USA: Elsevier Press. 2012; 383-466.
[9] Marc GA, Daniel M, Alfonso RP, Petr SAP. System and a constructive membrane-inspired DNA algorithm for solving the Maximum Clique Problem. *BioSystems.* 2007; 90(3): 687–697.
[10] Huang CY, Dong XJ, Long H. Solving Sorting Problem by P System. *Journal of Shanghai Jiaotong University.* 2008; 42(2): 206–208.