Referential Hierarchical Clustering Algorithm Based upon Principal Component Analysis and Genetic Algorithm

JUI-SHIN LIN, SHIAW-WEN TIEN, TUNG-SHOU CHEN[1], YUNG-HUNG KAO[1], CHIH-CHIANG LIN[1], YUNG-HSING CHIU[1]*
Graduate School of Technology Management, Graduate School of Computer Science and Information Technology[1]
Chung Hua University, National Taichung Institute of Technology[1]
707, Sec.2, WuFu Rd., Hsinchu, 300
Taiwan
JoeLin@mail.cvtc.gov.tw

Abstract: - Hierarchical Clustering (HC) is not designed to locate the leaf nodes in the tree structure, and therefore is not suitable to locate similarity relation on the sequence of the leaf nodes. In order to generate the similarity relation on tree structure diagram of HC, we proposed an improved solution in this paper; Referential Hierarchical clustering Algorithm (RHA). RHA is a combination of HC, Genetic Algorithm (GA) and Principal Component Analysis (PCA) to resolve the problem of traditional HC. PCA is a technique that reduces high-dimensional dataset to lower dimensions for analysis and reconstructs each data by a suitable linear combination of the principal components. These principal components are ordered by the amount of the variance which is explained in the original dataset. Therefore, RHA adopts GA to find the solution which has the same tree structure with HC and the most similar with the sequence of the samples sorted by increasing value of the first principle components. Experimental results show that the clustering result of RHA exposes the similarity relations between the leaf nodes and the clusters. RHA could be applied to any problem in HC and the generated tree diagram could assist researchers to compare and analyze each sample and find the relations between the clusters more easily and quickly.

Key-Words: - Hierarchical Clustering (HC), Principal Component Analysis (PCA), Genetic Algorithm (GA), Tree Structure Diagram, Similarity Relation

1 Introduction

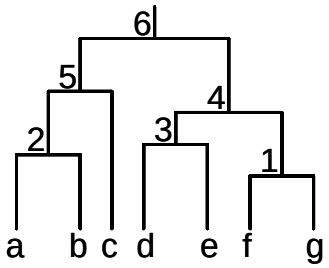
Clustering algorithms can be used to group samples into several clusters according to the difference of features. The samples which are within the same cluster are more similar than between different clusters. This technique is often used to resolve problems in pattern recognition [1], data mining [2], artificial intelligence [3] and so forth.

Hierarchical Clustering (HC) [4] has been widely used in various fields and can be subdivided into the agglomerative algorithm and the divisive algorithm [5]. The agglomerative algorithm regards each sample as a separate cluster, and then repeatedly merges the two most similar clusters into one cluster, until there is only one cluster. The divisive algorithm collocate all samples in one cluster, which is then divided into two smaller clusters, recurrently, until there is only one sample in each cluster. The algorithms based on HC like RIRCH [6], CURE [7], ROCK [8], CHAMELEON [9] and so on.

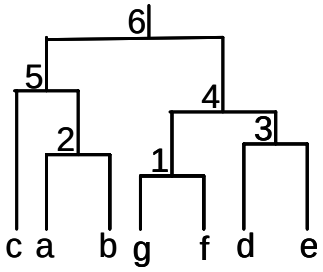
The result of HC can be expressed by a binary tree structure [10] which helps researchers to understand the clustering procedure easily and can accelerate the analysis of data. However, HC is not designed to locate the leaf nodes in the tree structure. As shown in Figs.1 (a) and (b), they are both the HC tree structures with the same dataset, but their leaf nodes sequences are different. Fig.1 (a) illustrates that leaf nodes f and g are merged first, because they are the most similar. Although node f is nearer node e than node g, it does not express that nodes f and e are more similar than nodes g and e. Thus there is not similarity relation in the leaf nodes sequence and researchers cannot identify the similarity relation at ease.

In order to resolve the problem of traditional HC, the Referential Hierarchical Clustering Algorithm (RHA) is proposed in this paper. RHA is a combination of HC, Genetic Algorithm (GA) [11] and Principal Component Analysis (PCA) [12]. PCA is a widely used technique for analyzing multivariate data. It can project the high dimensional data into lower dimensions, which can retain the characteristics which exist in high dimensional space. The principal components can represent the similarity relation in the sample. Therefore, it can be

used to change the sequence of the leaf nodes of HC to represent the similarity relationship in the sample. For this purpose, RHA adopts GA to find the solution which cannot modify the tree structure of clustering result and is the most similar with that of the first principal components.



(a) Tree structure 1



(b) Tree structure 2

Fig.1 The same tree structure diagrams of HC

The results obtained by the proposed algorithm in this paper show that the result of RHA is the most similar when sorted first by principal components. Hence RHA not only has the same utility functions with HC, it also provides the tree structure diagram which has the similarity relation.

2 Referential Hierarchical clustering Algorithm (RHA)

As shown in Fig.2, RHA consists of HC, PCA, and GA. In RHA, a chromosome of GA is a solution which expresses the same tree structure of HC, but has different leaf node sequences. The object of GA is to find a solution which is the most similar to the sequence of the sample sorted by increasing value of the first principle components. The above three parts of RHA are separately described as follows.

2.1 Hierarchical Clustering (HC) Algorithm

RHA employs the agglomerative algorithm in this work, since it is more popularly used than the other. Let *n* samples to be clustered, and the clustering process involves following steps:

Step 1: assign each sample to a cluster, thus we have *n* clusters and each contains one sample.

Step 2: calculate the similarities between two different clusters.

Step 3: Find the most similar cluster pair and merge them into one cluster.

Step 4: Repeat steps 2 through step 3 until all samples are clustered into one cluster.

The similarity calculation in Step 2 can be done by different methods [4,5], such as centroid linkage which calculates the similarity between the centroids of the two clusters. In this study, only results from the centroid linkage are shown due to space considerations.

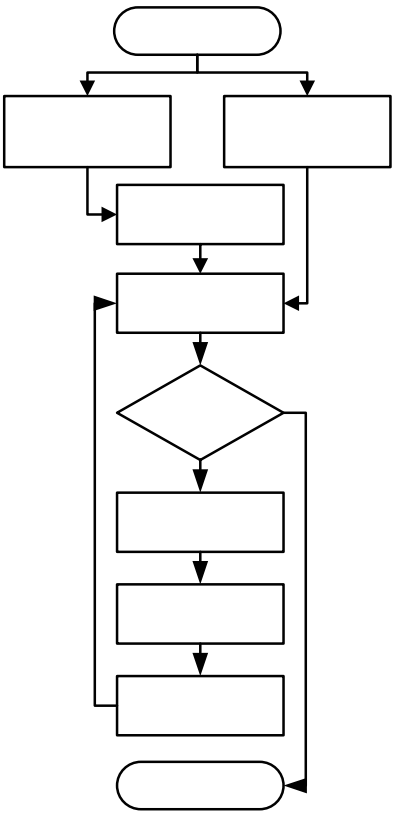


Fig.2 RHA flow chat

2.2 Principal Component Analysis (PCA)

PCA is a technique that reduces high-dimensional dataset to lower dimensions for analysis and then reconstructs each data by a suitable linear combination of the principal components. These principal components are ordered by the amount of variances used in explaining the original dataset. Therefore researchers usually use the most important principal components for the purpose of reducing the dimensionality of the data.

RHA refers to the first principal component to explain the greatest amount of variations in order to change the sequence of the leaf nodes of HC. The sample is sorted by increasing value of the first principle components and then this sequence will be

referred by the fitness function of GA to evaluate as in chromosomes.

2.3 Genetic Algorithm (GA)

Which sub-trees have to re-sequence for the leaf nodes to be the most similar with that of the samples sorted by increasing value of the first principle components? Flipping an internal-node may influence the others which were previously flipped, hence it is a difficult problem for the simple heuristic algorithm (i.e. Greediness); especially when processing a huge dataset. Therefore RHA adopts GA to find the best solution.

GA has been successfully applied to many fields, such as the traveling salesman problem (TSP) [14], knapsack problem [15], production scheduling [16], and so on. These subjects may be different; however it gives testimony that GA can process complex questions and finds the optimal or near optimal solutions. For a detail study on GA, readers can refer to [11].

The population size of GA is the amount of the chromosomes. At initialization, the population is assigned randomly, and to be evolved over generations. During each generation, GA creates a new population from the current population by applying three genetic operators: selection, crossover and mutation. The evolution of GA is simulated until a satisfying terminating condition is reached; meaning, the best solution is reached. The encoding of a chromosome and these three genetic operators are described as follows.

2.3.1 Encoding

As shown in Fig.3 (a), a chromosome is a binary string and each bit of a chromosome represents each internal-node in the tree structure. If a tree structure has n+1 leaf nodes, it will have n internal-nodes and the chromosomes will have n bit consequently.

As shown in Fig.3 (b), the bit values of $d1$, $d4$ and $d5$ are 1s, which represent that the node $d1$, $d4$ and $d5$ must be switched in the sub-trees. The other bits with values are 0s, will not be switched. Thus the sequence of the leaf nodes is modified to as illustrated in this figure.

2.3.2 Selection and Fitness Function

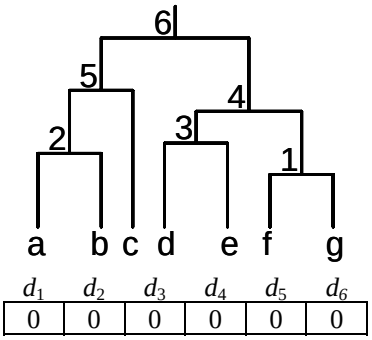
Each chromosome in the population is evaluated and is given a fitness value by the fitness function which is designed as follows:

$$f(S_j)=\sum_{i=1}^{n} | p_i - s_i | \qquad (1)$$

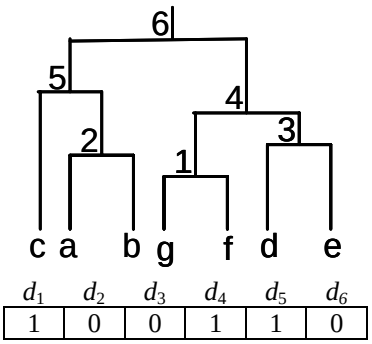
where i=1, 2, 3, …, n and j=1, 2, 3, …, m.

$f(S_j)$ represents the fitness value of the chromosome Sj in the population. The samples are sorted in the increasing value by the first principle components and are assigned serial numbers from 1 to n in the sorted sequence. Thus p_i is the serial number of the corresponding sample of the i-th first principle component. The solution S_j can be decoded to a corresponding sequence of the leaf nodes. The serial number of the i-th sample in the sequence of leaf nodes is s_i. Consequently, the leaf nodes and samples are sorted in increasing order sequence using the first principle components where the nodes are more similar and have a smaller fitness value; otherwise, the fitness value will be larger.

The selection operator used is the widely known roulette wheel strategy [11] which selects the chromosomes from the current population with probability inversely proportional to their fitness value for generating the next population.



(a) Original sequence of the leaf nodes



(b) Modified sequence of the leaf nodes

Fig.3 The examples of two chromosomes and corresponding tree structures

2.3.3 Crossover

The crossover operator is a strategy for sharing or exchanging the partial information between the chromosomes. RHA adopts the two-point crossover which swaps the segments of selected strings across two crossover points with the probability, and then two offspring chromosomes are generated to replace their parent chromosomes by increasing these two new chromosomes. For example, as shown in Fig.4, the two selected parent chromosomes are replaced by

the offspring chromosomes generated by performing a two-point crossover.

Parent 1	1	1	1	1	1	1	1
Parent 2	0	0	0	0	0	0	0
Crossover points			↓		↓		
Offspring 1	1	1	0	0	0	1	1
Offspring 2	0	0	1	1	1	0	0

Fig.4 An example of the two point crossover

2.3.4 Mutation

If the chromosomes in the population are too similar, GA may suffer from premature convergence. Therefore the mutation operator is introduced to avoid the final solutions to fall into a local optimum by randomly making slight changes to a chromosome. Thus GA has a chance to obtain better solutions that cannot be generated by a crossover operator.

As shown in Fig.5, the single point mutation is adopted by RHA, which selects a mutation point randomly and reverses the bit value selected in a chromosome chosen with a probability.

Before	1	1	1	1	1	1	1
Mutation Point				↓			
After	1	1	1	0	1	1	1

Fig.5 An example of the single point mutation

3 Experimental Results

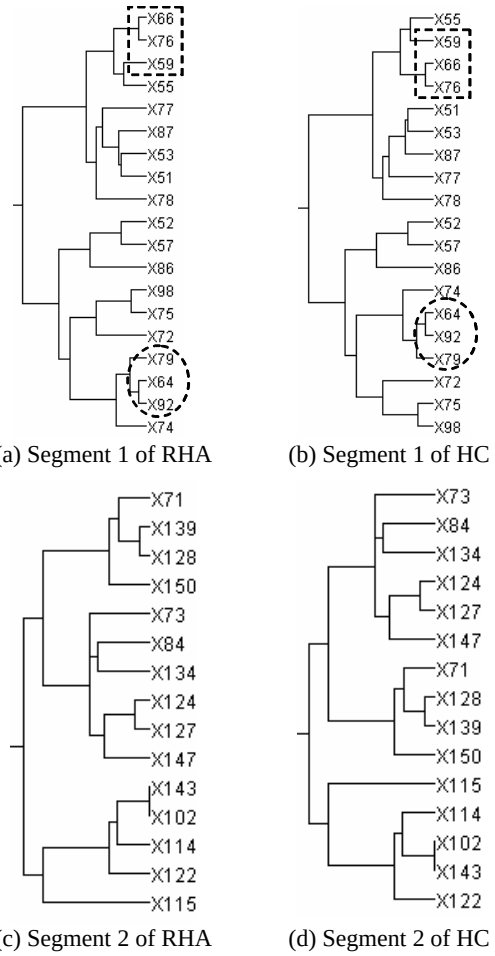
In this paper, we performed experiments by using the IRIS datasets taken from the UCI Machine Learning Repository [17]. The IRIS dataset is perhaps the best known and wildly used dataset in the clustering field. This dataset consists of three classes, fifty samples each and four numeric attributes defined for lengths and widths of sepals and petals. These three classes are referred to three different types of iris flower: Setosa, Versicolour and Virginica.

The result of RHA is compared with HC with centroids linkage. Since the tree structures are too large to demonstrate in this paper, only partial segments are illustrated in Fig.6. Partial sequences of the leaf nodes or samples are shown in Table1.

As shown in Table 1 and Fig.6, the sequences of the leaf nodes are different between RHA and HC. The leaf node sequence generated by RHA is more similar (samples are sorted by increasing value using the first principle components) in comparison to nodes in HC.

As Figs.6 (a) and (b) illustrate, the samples X64 and X92 are first merged together; and then merged with X79; and then with other samples to be agglomerated together in this way. Fig.6(c) and (d) also shows exactly the same situation. The above implies that the agglomerated processes of the two segments are not different. Hence the results generated by RHA and HC have the same tree structure.

Research on HC is concentrated on the similarity between each pair of the samples or clusters. This information is expressed by the heights of the sub trees. The similarity relationship between leaf nodes cannot be represented in original HC. As shown in Fig.6 (b), the similarity of the samples X59 and X76 are higher then that of the samples X59 and X66, therefore the sample X59 should not be near X66. RHA offers results which could represent the similarity relations between leaf nodes as illustrated in Figs.6 (a) and (c). Moreover, this improvement could be found everywhere in the result of RHA.



(a) Segment 1 of RHA (b) Segment 1 of HC

(c) Segment 2 of RHA (d) Segment 2 of HC

Fig.6 The result comparisons of RHA and HC in the partial segments

4 Conclusions

In this paper, we proposed an improved solution for generating the similarity relation on a tree structured diagram of HC. RHA adopts GA to find the solution which has the same tree structure with HC and the nearest similarity by sorting samples in increasing

value using the first principle components. Experimental results show that the clustering results of RHA enhance the similarity relations between the leaf nodes and the clusters.

Table 1 The comparison of the samples sequences

Methods	The samples sequence
Sorted by increasing value of the first principle components	X_{99}, X_{65}, X_{80}, X_{58}, X_{66}, X_{94}, X_{72}, X_{83}, X_{51}, X_{52}, X_{76}, X_{75}, X_{89}, X_{98}, X_{96}, X_{62}, X_{100}, X_{82}, X_{68}, X_{97}, X_{87}, X_{93}, …, X_6, X_{13}, X_8, X_{40}, X_{48}, X_{39}, X_{28}, X_{50}, X_{22}, X_7, X_{47}, X_{18}, X_{29}, X_3, X_{43}, X_{11}, X_{49}, X_1, X_{20}, X_5, X_{41}, X_{36}, X_{37}, X_{14}, X_{17}, X_{16}, X_{33}, X_{34}, X_{15}, X_{23}.
Referential Hierarchical clustering Algorithm (RHA)	X_{99}, X_{58}, X_{94}, X_{61}, X_{63}, X_{67}, X_{85}, X_{95}, X_{100}, X_{96}, X_{97}, X_{89}, X_{83}, X_{93}, X_{68}, X_{56}, X_{91}, X_{62}, X_{90}, X_{54}, X_{81}, X_{82}, ..., X_{47}, X_{49}, X_{11}, X_{41}, X_{05}, X_{40}, X_{08}, X_{29}, X_{28}, X_{01}, X_{18}, X_{50}, X_{32}, X_{21}, X_{37}, X_{44}, X_{27}, X_{24}, X_{23}, X_{15}, X_{19}, X_{06}, X_{17}, X_{33}, X_{34}, X_{16}, X_{42}.
Hierarchical Clustering (HC)	X_{16}, X_{15}, X_6, X_{19}, X_{17}, X_{33}, X_{34}, X_{23}, X_{12}, X_{25}, X_7, X_3, X_4, X_{48}, X_{13}, X_2, X_{46}, X_{10}, X_{35}, X_{38}, X_{26}, X_{30}, X_{31}, ... , X_{126}, X_{112}, X_{104}, X_{117}, X_{138}, X_{105}, X_{129}, X_{133}, X_{111}, X_{148}, X_{113}, X_{140}, X_{142}, X_{146}, X_{125}, X_{121}, X_{144}, X_{141}, X_{145}, X_{116}, X_{137}, X_{149}.

References:

[1] Chen, T.S., Lin, C.C., Chiu, Y.H., Lin, H.L. and Chen, R.C., A New Binary Classifier: Clustering -Launched Classification, *Lecture Notes in Artificial Intelligence*, Vol. 4114, 2006, pp. 278-283.

[2] Chen, T.S., Lin, C.C., Chiu, Y.H. and Chen, R.C., Combined Density- and Constraint-based Algorithm for Clustering, *The Proceedings of International Conference on Intelligent Systems and Knowledge Engineering*, Shanghai, China, 2006.

[3] Chen, T.S., Chen, R.C., Lin, C.C., Tsai, T.H., Li, S.Y., Liang, X., Classification of Microarray Gene Expression Data Using a New Binary Support Vector System, *The Proceedings of IEEE International Conference on Neural Networks and Brain*, 2005, pp. 485-489.

[4] Roiger, R.J. and Geatz, M.W., *Data Mining: A Tutorial-Based Primer*, Addison Wesley, 2003.

[5] Han, J. and Kamber, M., *Data Mining: Concepts and Techniques*, Morgan Kaufmann, 2000.

[6] Zhang, T., Ramakrishnan, R. and Livny, M., BIRCH: An Efficient Data Clustering Method for Very Large Databases, *The Proceedings of Conference on Management of Data*, 1996, pp. 103–114.

[7] Guha, S., Rastogi, R. and Shim, K., CURE: An Efficient Clustering Algorithm for Large Databases, *The Proceedings of Conference on Management of Data*, 1998, pp. 73–84.

[8] Guha, S., Rastogi, R. and Shim, K., ROCK: A Robust Clustering Algorithm for Categorical Attribute, *The Proceedings of Conference on Data Engineering*, 1999, pp. 512–521.

[9] Kaufman, L. and Rousseeuw, P.J., "*Finding Groups in Data: An Introduction to Cluster Analysis*," John Wiley and Sons, 1990.

[10] Saldanha, A.J., Java treeview-extensible visualization of microarray data, *Bioinformatics*, 2004, pp. 3246-3248.

[11] Holland, J.H., *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, Michigan, 1975.

[12] Jolliffe, I.T., *Principal Component Analysis*, Springer-Verlag, New York, 1986.

[13] Chen, T.S., Chen, Y.T., Lin, C.C., and Chen, R.C., A Combined K-Means and Hierarchical Clustering Method For Improving the Clustering Efficiency of Microarrary, *The Proceedings of International Symposium on Intelligent Signal Processing and Communications Systems*, 2005, pp. 405-408.

[14] Reinelt, G.., TSPLIB–A traveling salesman problem library, *ORSA J. Comput.* vol. 3, 1991, pp. 376-384.

[15] Sakawa, M. and Kato, K., Genetic algorithms with double strings for 0–1 programming problems, *European Journal of Operational Research*, vol.144, 2003, pp. 581–597.

[16] Chen, R.C., Chen, T.S., Feng, C.C., Lin, C.C. and Lin, K.C., Application of Genetic Algorithm on Production Scheduling of Elastic Knitted Fabrics, *Engineering and Applied Sciences*, vol. 1, no. 2, 2006, pp. 149-153.

[17] Newman, D.J., Hettich, S., Blake, C.L. and Merz, C.J., "*UCI Repository of Machine Learning Databases*," http://www.ics.uci.edu/ ~mlearn/MLRepository.html, 1998.