Thrifty Search of Interlocking Neighbor Grids for Enhancing Efficiency of Grid-Based Clustering

Cheng-Fa Tsai
Department of Management Information Systems,
National Pingtung University of Science and Technology,
Pingtung 91201, Taiwan
e-mail: cftsai@mail.npust.edu.tw

Yung-Ching Hu
Department of Management Information Systems,
National Pingtung University of Science and Technology,
Pingtung 91201, Taiwan
e-mail: m9956012@mail.npust.edu.tw

Abstract

Information technology has advanced with astonishing speed in the digital age, and the amount of data stored in databases is rapidly increasing. For data mining in large organizational databases, efficiency, accuracy and ease of use are essential. Thus, methods of retrieving useful information from large databases quickly and effectively need in-depth study. By using horizontal- and vertical-search methods to perform clustering diffusion, this grid-based algorithm reduces the cost of clustering and improves efficiency. Experimental comparisons show that the TING can function with a simple framework and can perform clustering for any graphics dataset. Performance comparisons also show that TING outperforms other well-known methods such as the IDBSCAN, GOD-CS and FARM algorithms, and its high accuracy and low run-time cost make it efficient and effective for data clustering in large databases.

Keywords: data mining, data clustering, grid-based clustering

Ⅰ. INTRODUCTION

Data mining methods have rapidly advanced in recent decades and now have many commercial, industrial and medical applications [3], [6]-[8]. Rapidly and accurately extracting concealed information from large databases is of primary importance in data mining. Therefore, many data clustering algorithms have been proposed in recent years to solve this data mining predicament.

Data clustering is a widely used data analysis technique in data mining. Data clustering methods can be classified as partitioning, hierarchical, density-based, grid-based or mixed. Partitioning methods group the data into K clusters by calculating the distance (e.g., the K-means [1]); these partitioning methods are characterized by high speed and simplicity, but they tend to be unstable and cannot filter noise. In contrast, density-based methods that identify the density of an area for data clustering (e.g., DBSCAN [2] and IDBSCAN [4]) can accurately identify any graphics and filter noise, but they tend to be time-consuming. In grid-based methods, the processing unit for determining the density of the grid-cell expansion is the grid-cell framework. However, although grid-based enable rapid clustering, they cannot run in a multi-dimensional space such as GOD-CS [5] and Farm [9].

This study proposes a thrifty search for interlocking neighbor grids (TING), which is a new grid-based clustering algorithm for increasing search efficiency in large databases. The TING applies a novel merger method to minimize the cost of searching for the neighbor grid by avoiding unnecessary searches, which improves efficiency. Experimental results prove that the proposed TING algorithm is a rapid and accurate clustering method.

The rest of the paper is organized as follows. Section Ⅱ surveys related works and summarizes the contributions of these methods. Section Ⅲ presents some connectional ideals of the algorithm. The details of TING algorithm are described in Section Ⅲ, and a performance study of TING is presented in Section Ⅳ. Finally conclusions are presented in Section Ⅴ.

Ⅱ. RELATED WORKS

2.1 Density-based algorithms

The DBSCAN algorithm is first approach for clustering by density-detecting. It depends on two arguments: ***Eps*** and ***MinPts***. ***Eps*** indicates a radius of search circle and ***MinPts*** represents a number of minimal neighbors in the search circle. These arguments are applied for examining the ε-neighbors each object contained. In the database, object P can be named ***core point*** if point P has at least ***MinPts*** neighbors and the distance has at most ***Eps*** from any object to point P. Subsequently, the search process is spreaded from point P to its neighbors. If the ε-neighbors of the point Q which is the neighbor of point P are under ***MinPts***, point Q would be named ***border point***. Employing this expansion, DBSCAN can accurately locate any arbitrary pattern. Because each expansion needs to examine all objects, the time complexity of DBSCAN would be also high when the size of database is large.

The IDBSCAN algorithm was developed to exclude the sampling on DBSCAN, and was presented in 2004 by Borah et al. Representative points on the circle create the Marked Boundary Objects (MBO). The MBO recognize the eight distinct objects in the neighborhood of an object. The points that are the closest to these eight positions are selected as seeds. Consequently, at most eight objects are labeled at once. IDBSCAN needs fewer seeds than DBSCAN.

2.2 GOD-CS

The GOD-CS clustering algorithm was proposed by Tsai, C.F. and Chiu, C.S. in 2010. GOD-CS employs grid-based clustering, neighborhood 8-square searching scheme and tolerance rate to lower the threshold with only two parameters, and speed up the execution time and enhance performance. Experimental results indicate that GOD-CS clusters groups effectively and efficiently in arbitrary patterns, and has very low execution time with a comparable correction rate.

2.3 FARM

FARM is one of efficient clustering algorithms that are concerned with grid, which is a grid-based algorithm using density-based concept for clustering data in data mining applications. In the FARM clustering approach, the number of separate clusters is required. It filters 98% of the noise, and the data set accuracy exceeds 99%. Processing 575,000 datasets takes less than one second.

Ⅲ. THE PROPOSED TING CLUSTERING ALGORITHM

The proposed TING algorithm has a somewhat similar framework but a very different extension method. Figure 1 depicts the framework. The ***DataSet*** is databases or dataset. The ***GridSize*** indicates the size of grid in terms of numbers or size. The ***MinPts*** expresses the smallest number of objects in a grid. The detailed steps of TING are as follows:

Step 1: Initialization all arguments.
Step 2: Use ***DataSize*** argument to partition the data space into matrix-like (grids)
Step 3: Determine the density of each grid in order to filter noise by ***MinPts*** argument.
Step 4: Sequentially load ***GridSeeds*** (effective-grid array), and check whether the grid is executed, if not, perform ***Horizontal_Search*** function. Otherwise, perform ***Vertical_Search*** function. This procedure is repeated until all ***GridSeeds*** are processed.

```
TING(DataSet , GridSize , MinPts)
    Initialization(GridSize, DataSet);
    FilterNoise(GridSize, MinPts);
    FOR i FROM 1 TO Effective_Grids.size DO
        Grid:= Effective_Grids.get(i);
        IF Grid.ClId=UNCLASSIFIED THEN
            GridSeeds.append(Grid);
            Grid.ClId:=ClusterID;
            FOR j FROM 1 TO GridSeeds.size DO
                GridSeed:=GridSeeds.get(i);
                IF GridSeed.checkProcess=UNPROCESS THEN
                    Horizontal _Search(GridSeed , ClusterID);
                ELSE
                    Vertical _Search(GridSeed , ClusterID);
                END IF
            END FOR
            ClusterID++;
    END FOR
END TING
```

Figure 1. Proposed TING algorithm.

The ***Horizontal_Search*** function only searches for individual grids to the left and right of ***GridSeed*** B. When an effective grid detected, the grid added to the seed table (***GridSeeds***). Otherwise, the left/right search stops. Figure 2 displays the conceptualization of Horizontal_Search function.

In Figure 3, the ***GridSeed*** is the current processing effective grid. The ***ClusterID*** depends on the group number. The ***GridSeed_RIGHT*** is the right of the ***GridSeed***. The ***GridSeed_LEFT*** is the left grid of the ***GridSeed***.

The ***Vertical_Search*** function searches for grids individually from the top to the bottom of ***GridSeed*** B. If the detected grid is effective, it is added to the seed table (***GridSeeds***). Figure 4 reveals the conceptualization of Vertical_Search function.

In figure 5, the ***GridSeed*** is the current processing effective grid. The ***ClusterID*** depends on the present group number. The ***GridSeed_UP*** is the up grid for the ***GridSeed***. The ***GridSeed_DOWN*** is the down grid for the ***GridSeed***.

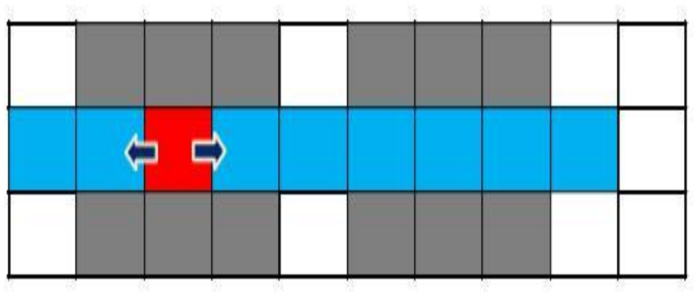


Figure 2. Conceptualization of Horizontal_Search function.

```
Horizontal_Search(GridSeed, ClusterID)
    WHILE GridSeed_RIGHT.GridId==EFFECTIVE DO
        GridSeed_RIGHT.checkProcess:=PROCESS;
        IF GridSeed_RIGHT.ClId= UNCLASSIFIED THEN
            GridSeeds.append(GridSeed_RIGHT);
            GridSeed_RIGHT.ClId:=ClusterID;
            GridSeed_RIGHT.next();
        END IF
    END WHILE
    WHILE GridSeed_LEFT.GridId==EFFECTIVE DO
        GridSeed_LEFT.checkProcess:=PROCESS;
        IF GridSeed_LEFT.ClId= UNCLASSIFIED THEN
            GridSeeds.append(GridSeed_LEFT);
            GridSeed_LEFT.ClId:=ClusterID;
            GridSeed_LEFT.next();
        END IF
    END WHILE
    GridSeed.checkProcess:=PROCESS;
    GridSeeds.append(GridSeed);
END
```

Figure 3. Horizontal_Search function.

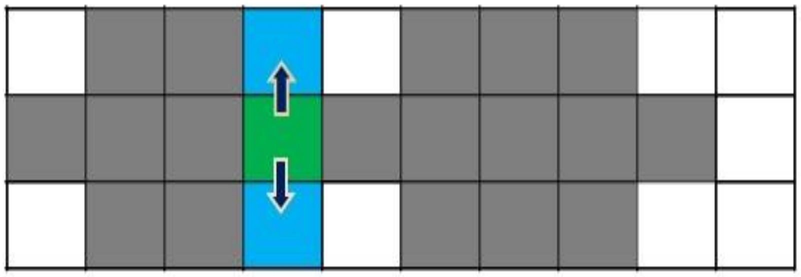
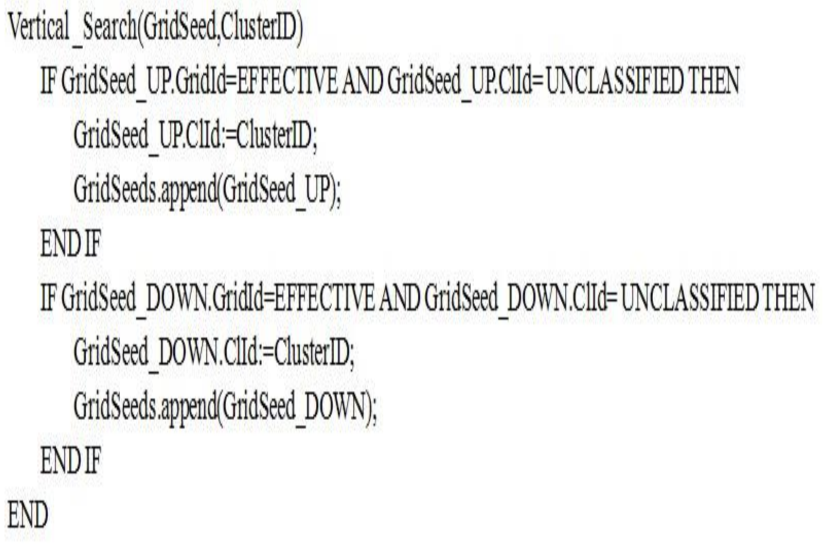


Figure 4. Conceptualization of Vertical_Search function



```
Vertical _Search(GridSeed,ClusterID)
    IF GridSeed_UP.GridId=EFFECTIVE AND GridSeed_UP.ClId=UNCLASSIFIED THEN
        GridSeed_UP.ClId:=ClusterID;
        GridSeeds.append(GridSeed_UP);
    END IF
    IF GridSeed_DOWN.GridId=EFFECTIVE AND GridSeed_DOWN.ClId= UNCLASSIFIED THEN
        GridSeed_DOWN.ClId:=ClusterID;
        GridSeeds.append(GridSeed_DOWN);
    END IF
END
```

Figure 5. Vertical_Search function.

Ⅳ. EXPERIMENT AND ANALYSIS

In this investigation, the proposed TING was implemented in a Java-based program using desktop computer with 2 GB RAM and an Intel 2.7GHz Dual-Core CPU on Microsoft Windows 7 professional Operational System. In the experiment, the results of the proposed TING were compared with IDBSCAN, GOD-CS and FARM. Additionally, we employ six synthetic datasets to test. All datasets has 575,000 objects in 2-D, and with 15% noise [10].

Figure 6 depicts the image of six synthetic datasets. Table 1 displays the experimental result from TING, IDBSCAN, GOD-CS and FARM. It shows comparisons of TING, IDBSCAN, GOD-CS and FARM using 575,000 data sets with 15% noise; ITEM 1 represents run-time cost (in seconds); ITEM 2 denotes the clustering correctness rate (%), while ITEM 3 indicates the noise filtering rate (%).The proposed TING had lowest run-time cost than others, and it also had high clustering correctness rate (CCR) and high noise filtering rate (NFR).

The clustering results in Figure 7 demonstrate that TING can process any graphics data, even complex graphics data such as those in data sets 3-6.

Although GOD-CS and FARM are fast, they waste search resources. In contrast, IDBSCAN, which is a density-based algorithm, is very slow, but it has highest accuracy in large databases, the speed difference between the grid-based algorithm and the density-based algorithm increases, but the difference in accuracy decreases. It is observed that the TING usually outperforms IDBSCAN, GOD-CS and FARM. As our best knowledge, the proposed TING algorithm is the fastest clustering approach in the world currently.

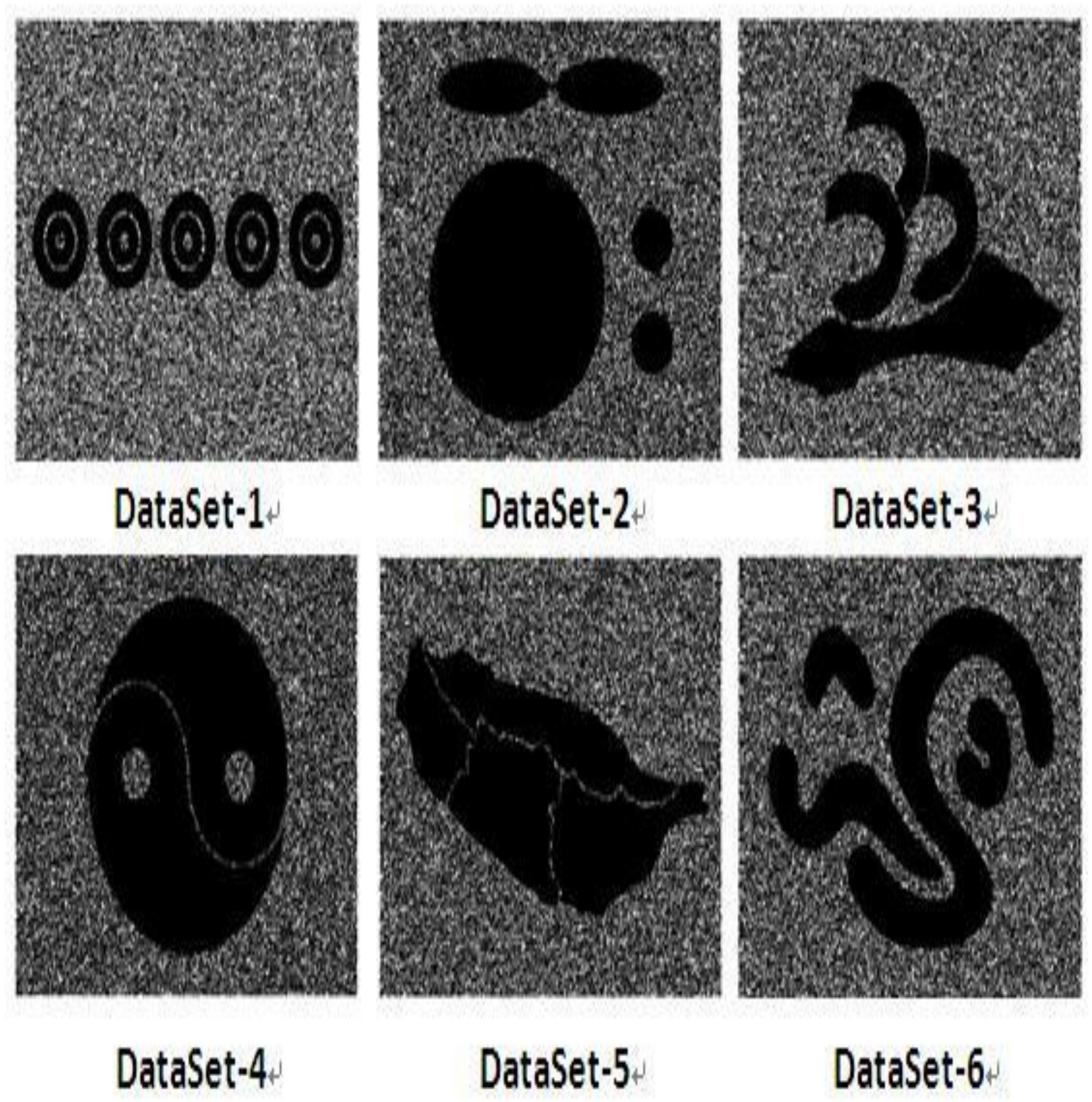


Figure 6. The original datasets.

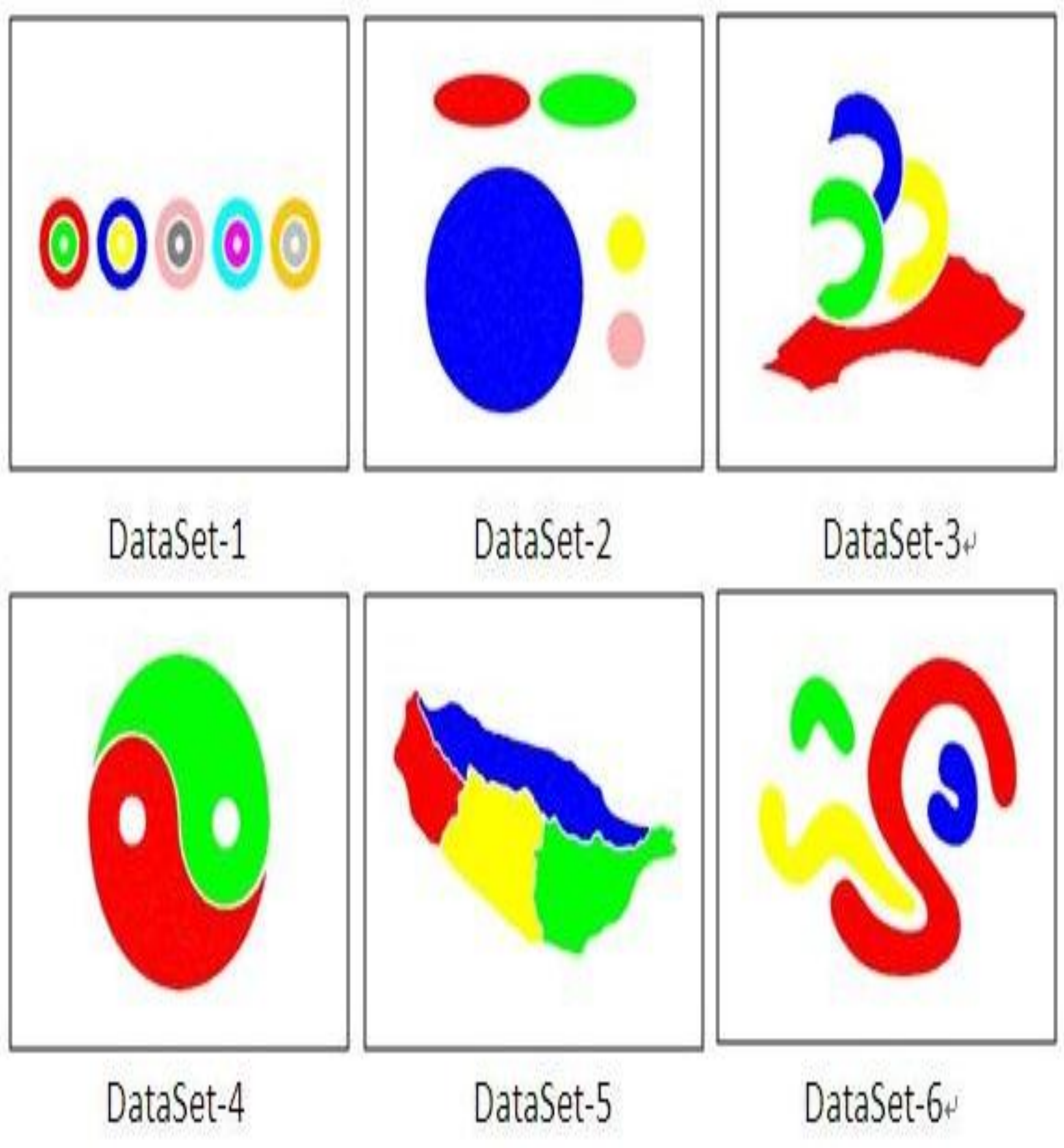


Figure 7. The clustering results with 575,000 objects using TING.

TABLE 1. COMPARISONS OF TING, IDBSCAN, GOD-CS AND FARM USING 575,000 DATA SETS WITH 15% NOISE.

Algorithm	**Item**	**DataSet-1**	**DataSet-2**	**DataSet-3**	**DataSet-4**	**DataSet-5**	**DataSet-6**
IDBSCAN	1	3128	3289	3274	3124	3282	3278
	2	99.99%	99.96%	99.99%	99.97%	99.99%	99.98%
	3	95.24%	94.25%	95.46%	93.37%	94.16%	94.21%
FARM	1	0.156	0.206	0.190	0.193	0.194	0.201
	2	99.92%	99.58%	99.57%	99.80%	99.79%	99.77%
	3	98.97%	99.27%	99.25%	99.10%	99.09%	99.12%
GOD-CS	1	0.330	0.351	0.367	0.351	0.359	0.361
	2	98.54%	98.69%	98.8%	98.84%	98.78%	98.82%
	3	99.43%	99.55%	99.84%	99.68%	99.63%	99.69%
TING	1	0.053	0.051	0.047	0.051	0.048	0.057
	2	99.58%	98.62%	99.61%	99.47%	99.32%	99.32%
	3	98.78%	99.61%	99.28%	99.24%	99.24%	99.24%

Ⅴ. CONCLUSION

Efficiency and accuracy is essential in the data mining field, but many algorithms are time-consuming or have low accuracy. Therefore, a highly efficient and accurate algorithm is needed. The proposed TING algorithm reduces repetitive searches of grids in the extend function, so it increases the efficiency of the grid-based cluster algorithm. Moreover, this algorithm is conceptually simple and requires only two arguments to operate. The experimental results show that the run-time of this method is much lower than that of other clustering algorithm, and the accuracy of this method is also within 99%.Thus, the method performs well in terms of both efficiency and accuracy.

ACKNOWLEDGMENTS

The author would like to thank the National Science Council of Republic of China for financially supporting this research under contract no. NSC 98-2221-E-020-029 and NPUST 98-005.

REFERENCES

[1] McQueen, J. B., “Some Methods of Classification and Analysis of Multivariate Observations,” Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability, pp. 281-297, 1967.

[2] Martin Ester, Hans-Peter Kriegel, Jorg Sander,Xiaowei Xu, “A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise,” Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining, pp. 226-231, 1996.

[3] Agrawal, R., Gehrke, J., Gunopulos, D., Raghavan, P., “Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications.” Proc. ACM SIGMOD International Conference on Management of Data, pp. 94–105. ACM Press, Seattle, Washington, 1998.

[4] B. Borah,D K Bhattacharyya, “An Improved Sampling-Based DBSCAN for Large Spatial Databases,” Proceedings of international conference on intelligent sensing and information processing, pp. 92–96, 2004.

[5] Cheng-Fa Tsai and Chien-Sheng Chiu, “GOD-CS: A New Grid-Oriented Dissection Clustering Scheme for Large Databases,” In: R. Huang et al. (Eds.): ADMA 2009, LNAI, vol. 5678, pp. 302–313, 2009.

[6] Tsai, C.F., Yen, C.C., “G-TREACLE: A New Grid-Based and Tree-Alike Pattern Clustering Technique for Large Databases,” In: Washio, T., Suzuki, E., Ting, K.M., Inokuchi, A. (eds.) PAKDD 2008. LNCS (LNAI), vol. 5012, pp. 739–748. Springer,Heidelberg, 2008.

[7] Tsai, C.F., Yen, C.C., “ANGEL: A New Effective and Efficient Hybrid Clustering Technique for Large Databases,” In: Zhou, Z.-H., Li, H., Yang, Q. (eds.) PAKDD 2007. LNCS (LNAI), vol. 4426, pp. 817–824. Springer, Heidelberg, 2007.

[8] Wang, T.P., Tsai, C.F., “GDH: An Effective and Efficient Approach to Detect Arbitrary Patterns in Clusters with Noises in Very Large Databases,” Degree of master at Pingtung University of Science and Technology, Taiwan, 2006.

[9] Cheng-Fa Tsai, Kuei-Sheng Lee, “FARM: A New Efficient and Effective Data Clustering Algorithm,” 9th WSEAS International Conference on Multimedia Systems & Signal Processing, hangzhou, China, PP. 253-258, May 2009.

[10] Cheng-Fa Tsai, Heng-Fu Yeh, Jui-Fang Chang, Ning-Han Liu, “PHD: an efficient data clustering scheme using partition space technique for knowledgediscovery in large databases”, Vol. 33, pp. 39-53, June 2010.

可減少連鎖網格搜尋的高效率網格式叢集分析技術

蔡正發
國立屏東科技大學資訊管理系
cftsai@mail.npust.edu.tw

胡永慶
國立屏東科技大學資訊管理系
m9956012@mail.npust.edu.tw

摘要

在今日的數位時代中，資訊科技的蓬勃發展導致組織擁有大量的資訊，是以使組織內部的資料庫如何在運用資料探勘技術後使挖掘之知識更有效率及更具正確性是重要的研究議題。本研究提出一個以網格式分群為基礎的資料叢集演算法稱為為 TING，此法係基於將資料集切割成以網格為處理單位，利用最小包含點參數將網格分為有效格與無效格，並依序讀取有效格進行判斷挑選搜尋合併方式。而搜尋合併方式有兩種，分別為左右循序搜尋及上下搜尋，假若該有效格未進行過左右循序搜尋，則進行左右循序搜尋；反之，則進行上下搜尋，透過此法可大量減少傳統網格擴散方式之重複搜尋網格次數，進而提升叢集效率。經由模擬實驗結果顯示，TING 可以基於一個簡單的架構下執行資料分群，同時可以大幅度的減少所需花費的時間成本。與各個國際現存的著名網格式分群演算法比較，TING 於執行速度上確實比 DBSCAN、IDBSCAN、GOD-CS 等分群演算法快速，且在分群正確率與雜訊濾除率上都有優異的表現（均落於 98%~99%之間）。經由實驗結果可證明本研究所提出的 TING 資料分群演算法同時具備了效率性及正確性。就筆者所知，TING 分群演算法應是目前全球最快速的網格式分群技術。

關鍵詞：資料探勘、資料分群、網格式分群