A Hybrid Approach to Cluster Detection

Samir Tout, Junping Sun, and William Sverdlik

***Abstract*—Recent technological advances require computer algorithms that can effectively analyze and classify data on a large scale that was unachievable just a few years ago. For instance, in response to a query, commercial search engines routinely consider web pages amounting into billions while genomic searches may deal with a search space of a similar or even higher magnitude. Clustering algorithms are an ideal choice to quickly categorize data; they are conceptually simple and require little background knowledge. Many clustering algorithms have been introduced in recent decades; but each approach brought along new challenges to consider, such as outlier handling, detection of arbitrary shaped clusters, processing speed, and dependence on user-supplied parameters. PYRAMID, or parallel hybrid clustering using genetic programming and multi-objective fitness with density, is a clustering algorithm that we introduced in a previous research. It addresses several of the above challenges by using a combination of data parallelism, a form of genetic programming, and a multi-objective density-based fitness function. This paper summarizes some of the characteristics of PYRAMID along with experiments that were performed on multiple challenging datasets. Empirical results derived from these experiments are presented and future directions are proposed.**

***Index Terms*—**Data Mining, Clustering, Genetic Programming, Density, Parallelism.

I. Introduction

Clustering algorithms are frequently employed in situations where large amounts of data must be categorized and little background knowledge is available. Such applications include document clustering, which has become a major focus of search engine technology [7], gene processing in bioinformatics, a field that has grasped considerable attention in the last decade [4], and pattern recognition [10]. Recent decades have witnessed several clustering approaches that introduced new challenges, including outlier handling, detection of arbitrary shaped clusters, processing speed, and dependence on user-supplied parameters. In [18], we introduced PYRAMID, or *Parallel hYbrid clusteRing using genetic progrAmming and Multi-objective fItness with Density*, which uses a combination of data parallelism, a form of genetic programming (GP), and multi-objective fitness function to remedy some of these challenges. PYRAMID employs data parallelism to improve performance by dividing the clustering data among multiple processors. It attempts to detect arbitrary shaped clusters by leveraging the flexible representational power of genetic programming and addresses outlier detection by employing a density based fitness function. The experiments conducted in [18], which used data sets of various sizes and irregular cluster shapes have demonstrated positive results. These results are used to compare cluster and outlier detection between PYRAMID and existing known algorithms such as BIRCH [20], CURE [6], DBSCAN [5], and NOCEA [13].

This paper borrows from [18] and provides a brief introduction to the PYRAMID algorithm. It also elaborates on its detection capabilities by summarizing the results of several experiments on various data sets that present special challenges such as variable shapes, extensive outliers, and clusters with holes, sharp contours, and pointy extremities.

The rest of this paper is organized as follows. Section 2 provides a listing of related literature work. Section 3 introduces key concepts in this study. Section 4 provides a brief overview of PYRAMID. Section 5 presents some of the experiments as well as a description of the data sets. Finally, Section 6 states the conclusion of this research and future directions.

II. Related Work

Several clustering algorithms were introduced in the last two decades, which addressed some of the challenges mentioned above. For instance, CURE [6] used data samples as well as an interesting shrinking mechanism to detect outliers. BIRCH [20] employed data summarization for best detection on circular clusters. DBSCAN [5] used density for better cluster detection. RBCGA [12] utilized genetic algorithm to discover rectangular cluster shapes. NOCEA [13], a

Samir Tout is a consultant with Keane, Inc., 24901 Northwestern Hwy, Southfield, MI 48075 and an adjunct professor at the Department of Computer Science, Eastern Michigan University, Ypsilanti, MI, 48197 (e-mail: samtout@gmail.com, stout@emich.edu).
Junping Sun is a professor at the Graduate School of Computer and Information Sciences, Nova Southeastern University, 3301 College Avenue, Fort Lauderdale, Florida 33314, USA (e-mail: jps@nova.edu).
William Sverdlik is an associate professor at the Department of Computer Science, Eastern Michigan University, Ypsilanti, MI, 48197 – wsverdlik@emich.edu.

(Advance online publication: 15 August 2007)

successor of RBCGA, provided better detection but mostly resulted in coarse detections [13]. The next two sections borrow directly from [18] in the subsequent definitions and description of the PYRAMID approach.

III. Definitions

This section briefly introduces terms and concepts that are pertinent to the PYRAMID algorithm. For simplicity, the rest of this study focuses on two-dimensional data space as in [18] and leaves higher dimensions for future research. The reader is encouraged to refer to [18] for further details.

A *minimum bounding rectangle* (MBR) is the smallest rectangular area in the data space that contains all points in a specific data set [11]. *Binning* within an MBR is the division of the *x* and *y* axes, respectively, into t_x and t_y non-overlapping segments, called *bins*, having the same lengths per dimension. The intersections of the bin lines, or quantization, construct a 2-dimensional grid that divides the MBR into contiguous non-overlapping 2-dimensional cells.

A Rule r is a rectangular sub-region of the MBR that contains one or more contiguous cells. This study does not allow overlapping rules, i.e. sharing common cells, within the same solution. An *Individual I* is formed by the union of rules within the MBR. The size of an individual *size(I)* is the number of rules in *I*. Refer to [18] for further details about the cardinality, volume, and density of a cell, rule, and individual. *Geometric Division* is an algorithm that divides the data space into quadrants, each containing a data subset formed by the data points that belong to its constituent cells. The details of this algorithm are outlined in [18] and exemplified in Fig. 1.

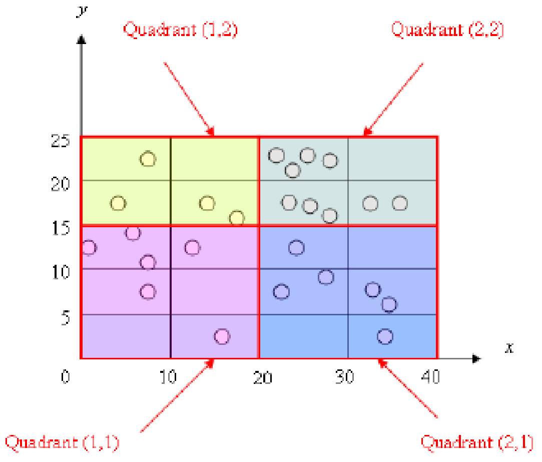

Fig. 1. Sample geometric division.

IV. The Approach

The PYRAMID algorithm, summarized in Fig. 2, is a multi-step hybrid approach that utilizes the above concepts. It is further described in the following sections.

```
Master Processor
1. Initiate binning.
2. Perform geometric division.
3. Send each subset to a different slave.
4. Receive p resulting subsets of discovered
   data points from p slaves. Determine
   cells that contain returned points.
5. Merge returned cells into global solution
   that labels every cell with a cluster.
Slave Processor
1. Receive data subset P from master.
   Perform quantization on local data.
2. Run genetic program on the local data
   points in P (on current slave processor).
3. After algorithm finishes, send points in
   discovered cells to master processor.
```

Fig. 2. Master and slave roles in PYRAMID.

A. Master-Slaves Communication

The first step in PYRAMID is executed by the Master processor, which performs the geometric division, forming quadrants as groups of cells. Subsequently, the master processor sends each quadrant's data subset to a separate slave processor that executes the following genetic program.

B. Genetic Program

In this study, a genetic program is used that encodes every individual, *I*, as a tree having leaf nodes representing *I*'s constituent rules. This representation offers more flexibility than genetic algorithm-based bit-strings [10], as demonstrated by the example in Fig. 3, which represents individual I_1 from Fig. 4. As in standard genetic programming, the internal nodes correspond to the functions that apply to the leaf nodes [10]. In this study, union is the only function employed. It symbolizes that the individual is formed as a combination of its constituent rules.

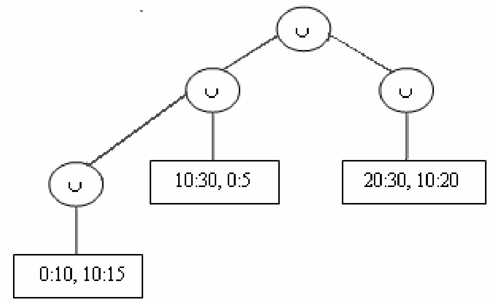

Fig. 3. Tree representation of Individual I_1 in Fig. 4.

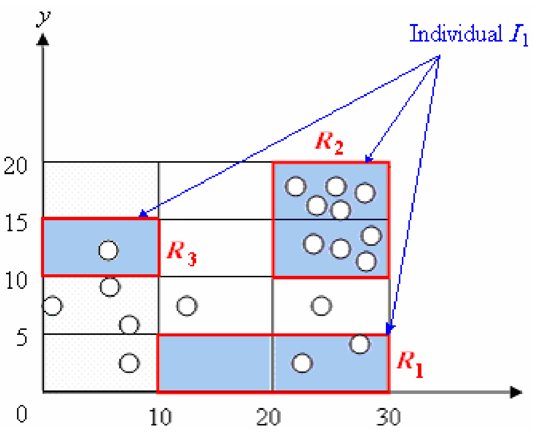

Fig. 4. Rules for individual in Fig. 3.

1) Genetic Operators

The main genetic operators used by PYRAMID are crossover, smart mutation, architecture altering (also called structural), and repair. This section provides a brief overview of these operators, but the reader is referred to [18] for further details.

Crossover acts at the rule level by swapping rules between individuals thus producing two new individuals. *Smart mutation* has two flavors: *enlarge mutation*, which attempts to add cells in dense neighborhoods and *shrink mutation*, which takes out cells with respect to a specific dimension. Mutation always produces one new individual. *Architecture altering* adds a new rule to an individual or deletes an existing one from it. An operator was added in [18], called *repair*, which reshapes overlapping rules into new ones that align better with the distribution of the data points. This is demonstrated by the example in Fig. 5 where the frame depicts the area covered by the original rule.

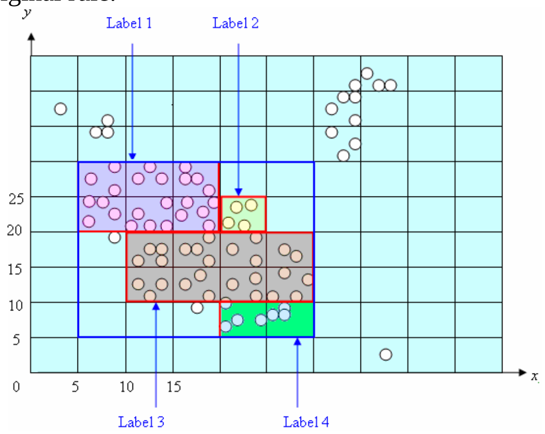

Fig. 5. Sample PYRAMID repair operation.

2) Fitness Function

PYRAMID uses a fitness function that focuses on three main factors to achieve good solutions: coverage, dense neighborhoods, and smaller individuals by means of parsimony pressure [18]. Therefore, PYRAMID's fitness function, incorporates the following three main objectives, as shown in (1).

$$Fitness(I) = \frac{F_{coverage}(I) \times F_{density}(I)}{F_{size}(I)} \qquad (1)$$

3) Selection Operator and Elitism

This study adopts a selection operator that is based on tournament selection with a tour size of three [3]. It also implements one-individual elitism, whereby in every iteration, the best performer is carried over to the next generation [2].

4) Main Algorithm

The GP that is run on each slave processor is summarized in Fig. 6. After each operator is applied, the fitness of resulting individuals is evaluated.

```
t = 0
Initialize population t
Evaluate population t
While (not termination condition)
 Begin
  t = t + 1
  s = selection from population t-1
  c = crossover 2 individuals in t
  m = smart mutation
  a = architecture-altering
  e = elitism
  Evaluate(fitness) population t
 End
```

Fig. 6. Serial GP algorithm.

C. The Merge Phase

In this final phase of the PYRAMID algorithm, the discovered points are reported back to the master, which traverses their associated cells, assigning them cluster labels based on their neighborhoods. The merge algorithm was discussed in details in [18].

V. Experiments

Our previous study [18] included multiple experiments that tested the ability of PYRAMID to detect clusters of arbitrary shapes, to dynamically determine the number of clusters, to achieve speedup using parallelism, its independence of the order of input, and its handling of outliers. Another study that we conducted [19] added further experiments using a new challenging data set and proved the resilience of PYRAMID to user-supplied parameters. This study adds more experiments that test data sets bearing other aspects, such as special contours and curvatures. The rest of this section revisits some of the experiments from [18], [19], as well as the ones mentioned above.

In [18], the experiments were run over existing two-dimensional data sets that were used by other algorithms like NOCEA, CURE, DBSCAN, and

RBCGA. Table 1 shows a list of these data sets. In addition, a new data set called DS5, which we introduced in [19], is also included in this table.

TABLE 1. DATA SETS USED IN PYRAMID EXPERIMENTS.

DATA SETS	POINTS	CLUSTERS
DS1	8,000	6
DS2	10,000	9
DS3	100,000	6
DS4	1,120	3
DS5	100,000	100

Fig. 7, Fig. 8, and Fig. 9 provide a comparison between the PYRAMID detection of DS1, DS2, and DS3 against NOCEA, CURE, and DBSCAN. It is evident that PYRAMID provides smoother detection than NOCEA, better discovery and outlier handling than CURE and DBSCAN [9]. Fig. 10 demonstrates a smoother detection by PYRAMID than RBCGA.

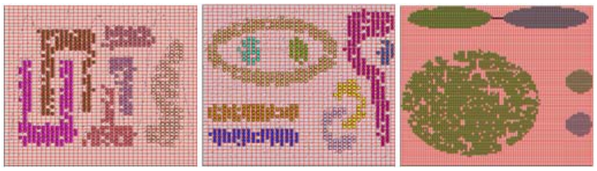

Fig. 7. PYRAMID cluster discovery.

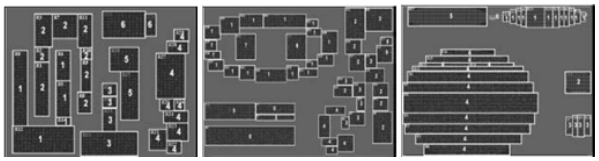

Fig. 8. NOCEA cluster discovery [13].

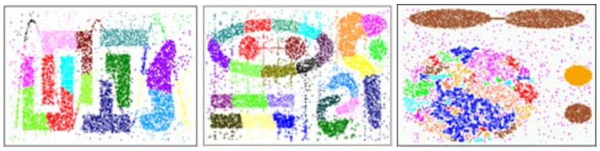

Fig. 9. DS1, DS2, DS3 by CURE and DBSCAN [9].

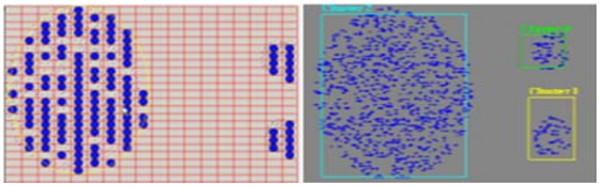

Fig. 10. DS4 by PYRAMID versus RBCGA [12].

The independence of PYRAMID on the order of data input was also demonstrated in [18], as shown Fig. 11, which depicts the detection of the same data set with a different data order. It is evident that both detections are similar, thus demonstrating the independence of PYRAMID on the order of input.

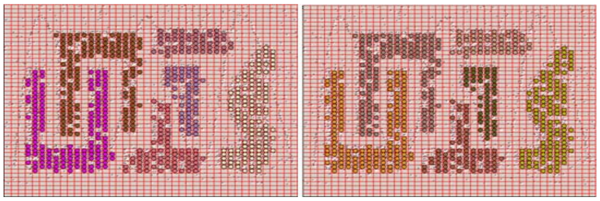

Fig. 11. Detection with different data order.

Other experiments were conducted in [18] to evaluate the improvements in speed that PYRAMID achieved from serial to parallel with four and sixteen slave processors, for data sets DS1, DS2, and DS3. The results showed considerable speedup improvements that ranged from 1.8 to 6.43. The reader is encouraged to refer to [18] for additional details.

Further experiments were performed in [19] that evaluate the performance of PYRAMID with a more challenging data set, referred to as DS5, which contains one hundred clusters that are close and surrounded with a considerable amount of outliers. In [19], we also evaluated PYRAMID's independence on user-supplied parameters.

Independence of PYRAMID on user parameters:

This experiment was performed in [19] to evaluate the impact of modifying different parameters on the outcome of the PYRAMID algorithm. The results have shown that detection remained fairly similar even when crucial parameters such as the genetic program population size, number of rules per individual, and the genetic operator percentages. This is demonstrated in Fig. 12 where these parameters were changed and the results are compared to the original PYRAMID run for DS2 shown in Fig. 7.

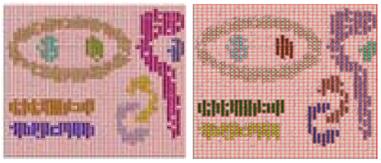

Fig. 12. PYRAMID with different parameters on DS2.

Experiments with other data sets:

This study adds more experiments using other data sets, obtained from [1], which do not contain a large number of points but rather present different challenges such as clusters with holes, sharp contours, and pointy extremities. As demonstrated in the next set of figures, PYRAMID shows fairly good detection of clusters in these data sets. The left side of all these figures is the actual data as drawn by gnuplot.

It is noticeable in Fig. 13 that PYRAMID captured the shape of DS6 data distribution but missed a small detail on the top right corner of the cluster. A similar scenario is encountered in Fig. 14 where the detection is fairly similar to the actual data distribution with some subtle differences. This demonstrates how PYRAMID detects a cluster that contains a hole, in this case with an oval shape.

TABLE 2. DATA SETS USED IN PYRAMID EXPERIMENTS.

DATA SETS	POINTS	CLUSTERS
DS6	459	1
DS7	388	1
DS8	857	1
DS9	489	1
DS10	4961	1

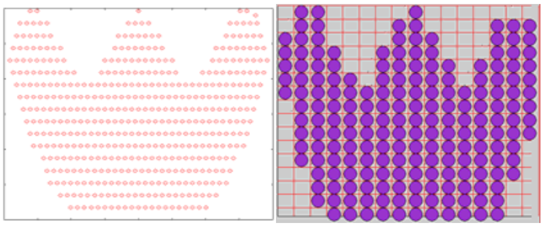
Fig. 13. DS6 detection using PYRAMID.

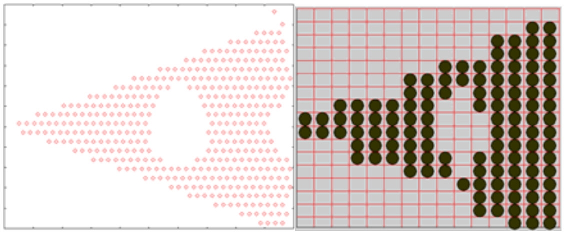
Fig. 14. DS7 detection using PYRAMID.

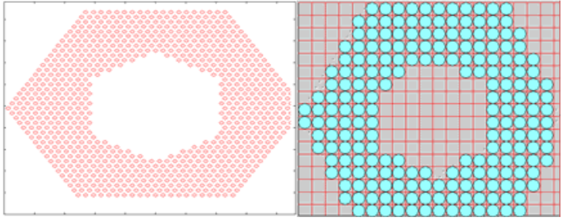
Fig. 15. DS8 detection using PYRAMID.

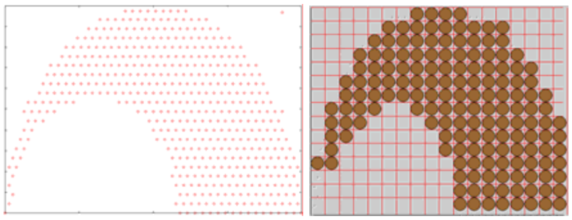
Fig. 16. DS9 detection using PYRAMID.

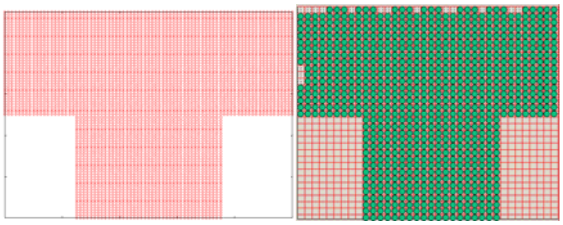
Fig. 17. DS10 detection using PYRAMID.

Fig. 15 shows another close detection by PYRAMID for a cluster with a hexagon-shaped hole. Fig. 16 demonstrates the detection of sharp curvatures while Fig. 17 shows the detection of sharp rectangular shapes, referred to as T-cells in [1]. It is worth noting in this last figure that there are some missing detections, which are mostly due to the non-deterministic nature of the GP-based algorithms that may result in a slightly different detection with every run, as demonstrated in more than one experiment in [18].

VI. CONCLUSION AND FUTURE WORK

In [18], we introduced a novel approach to clustering large data sets, called PYRAMID. It employed a hybrid combination of GP's global search and strong representational capabilities along with a powerful density-aware multi-objective fitness function as well as data parallelism to achieve speedup. The experiments that were performed in [18] used renowned data sets that were tested by other algorithms like CURE, BIRCH, and NOCEA. They demonstrated that PYRAMID detects clusters of arbitrary shapes, is mostly immune to outliers, and does not depend on the order of data input. In addition, its inherent data parallelism allows it to improve performance.

In another study [19], we also exercised the ability of PYRAMID to detect a more challenging dataset, DS5, which was employed in previous well known clustering research [15]. The results showed a performance by PYRAMID that was slightly better than WaveCluster. Another experiment that we performed also attested to the independence of PYRAMID on user-supplied parameters.

This study added other data sets that present different types of challenges such as clusters with oval and hexagon shaped holes, as in DS7 and DS8, sharp contours, as in DS9 and DS10, and pointy extremities like DS6 and DS7. As seen in the above figures, PYRAMID has shown that it is able to detect their clusters to a good degree.

One potential avenue for future research is to explore the performance of PYRAMID through speedup with higher dimensions. Other avenues include exploring the use of rules with variable shapes, not strictly rectangular, and using other forms of parallelism.

REFERENCES

[1] Berkardt, (2005). Datasets. Retrieved December 1, 2006 from http://www.csit.fsu.edu/~burkardt/datasets

[2] Berkhin, P. (2002). Survey of clustering data mining techniques. *Accrue Software*. Retrieved February 28, 2005 from http://www.ee.ucr.edu/~barth/EE242/clustering_survey.pdf

[3] Davis, L. (1991). *Handbook of Genetic Algorithms*. New York, NY: Van Nostrand Reinhold.

[4] Dettling, M. & Bühlmann, P. (2002). Supervised clustering of genes. *Genome Biology*, *3*(12), 39-50.

[5] Ester, M., Kriegel, H.P., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, Portland, Oregon,* 226-231.

[6] Guha, S., Rastogi, R., & Shim, K. (1998). CURE: An efficient clustering algorithm for large databases. *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data, Seattle, WA,* 73-84.

[7] Han, J. & Kamber, M. (2001). *Data Mining, Concepts and Techniques.* San Francisco, CA: Morgan Kaufmann.

[8] Hinneburg, A. & Keim, D.A. (1998). An efficient approach to clustering in large multimedia databases with noise. *Proceedings of the Fourth International Conference on Knowledge Discovery in Databases, New York, NY,* 58-65.

[9] Karypis, G., Han, S., & Kumar, V. (1999). Chameleon: A hierarchical clustering using dynamic modeling. *IEEE Computer: Special Issue on Data Analysis and Mining, 32*(8), 68-75.

[10] Koza, J.R. (1991). Evolving a computer program to generate random numbers using the genetic programming paradigm. *Proceedings of the Fourth International Conference on Genetic Algorithms, La Jolla, CA,* 37-44.

[11] Ohsawa, Y. & Nagashima, A. (2001). A spatio-temporal geographic information system based on implicit topology description:STIMS. *Proceedings of the Third International Society for Photogrammetry and Remote Sensing* (*ISPRS*) *Workshop on Dynamic and Multi-Dimensional Geographic Information System, Thailand,* 218-223.

[12] Sarafis, I., Zalzala, A., & Trinder, P. (2002). A genetic rule-based data clustering toolkit. *Proceedings of the 2002 World Congress on Evolutionary Computation, Honolulu, USA,* 1238-1243.

[13] Sarafis, I., Zalzala, A., & Trinder, P. (2003). Mining comprehensive clustering rules with an evolutionary algorithm. *Proceedings of the Genetic and Evolutionary Computation Conference, Chicago, USA,* 1-12.

[14] Scott, D. (1992). *Multivariate Density Estimation: Theory, Practice and Visualization.* New York, NY: John Wiley and Sons.

[15] Sheikholeslami, G., Chatterjee, S., & Zhang, A. (1998). WaveCluster: a multi-resolution clustering approach for very large spatial databases. *Proceedings of the 24th Intl. Conf. on Very Large Data Bases, New York,* NY, 428-439.

[16] Silverman, B.W. (1986). *Density Estimation for Statistics and Data Analysis.* London, UK: Chapman and Hall.

[17] Sturges, H. (1926). The choice of a class-interval. *Journal of the American Statistical Association*, *21*(1), 65–66.

[18] Tout, S., Sverdlik, W., & Sun, J. (2006). Parallel hybrid clustering using genetic programming and multi-objective fitness with density (PYRAMID). *Proceedings of the 2006 International Conference on Data Mining* (*DMIN'06*), *Las Vegas, NV, USA,* 197-203.

[19] Tout, S. & Sverdlik, W. , & Sun, J. (2007). Cluster detection with the PYRAMID algorithm. *Proceedings of the 2007 IEEE Symposium on Computational Intelligence and Data Mining, Hawaii, USA.*

[20] Zhang, T., Ramakrishnan, R., & Livny, M. (1996). BIRCH: An efficient data clustering method for very large databases. *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data, Montreal, Canada,* 103-114.