Journal of Universal Computer Science, vol. 11, no. 8 (2005), 1440-1453
submitted: 10/3/05, accepted: 5/5/05, appeared: 28/8/05 © J.UCS

Resource-aware Mining of Data Streams

Mohamed Medhat Gaber
(Centre for Distributed Systems and Software Engineering
Monash University
900 Dandenong Rd, Caulfield East, VIC3145, Australia
Mohamed.Medhat.Gaber@infotech.monash.edu.au)

Shonali Krishnaswamy
(Centre for Distributed Systems and Software Engineering
Monash University
900 Dandenong Rd, Caulfield East, VIC3145, Australia
Shonali.Krishnaswamy@infotech.monash.edu.au)

Arkady Zaslavsky
(Centre for Distributed Systems and Software Engineering
Monash University
900 Dandenong Rd, Caulfield East, VIC3145, Australia
Arkady.Zaslavsky@infotech.monash.edu.au)

Abstract: Mining data streams has raised a number of research challenges for the data mining community. These challenges include the limitations of computational resources, especially because mining streams of data most likely be done on a mobile device with limited resources. Also due to the continuality of data streams, the algorithm should have only one pass or less over the incoming data records. In this article, our Algorithm Output Granularity (*AOG*) approach in mining data streams is discussed. AOG is a novel adaptable approach that can cope with the challenging inherent features of data streams. We also show the results for AOG based clustering in a resource constrained environment.

Keywords: data mining, data streams, clustering, and resource-aware computing
Category: H

1 Introduction

Knowledge extraction from data streams has attracted attention in recent years. The continuous high-speed generation of data records from sensors, web click streams, and stock market information has created new challenges and open research issues for the data mining community [22], [23]. The intuitive solution is the development of one-pass or less techniques that can result in acceptable approximation mining results in a resource constrained environment; represented by low computational power of mobile devices and limited bandwidth of wireless networks. This represents the typical processing environment for data streams.

Different mathematical and algorithmic methods have been proposed to be used for data stream processing. Sampling and projection have been used to cope with the high data rate of data streams. Sampling refers to the process of selecting data items from a data stream according to a statistical measure. Projection is used for dimensionality reduction using sketching techniques [22]. Group testing, tree method, robust approximation and exponential histograms have been used as algorithmic techniques for reducing space and time complexity. [2], [22], [23].

Querying and mining data streams have been studied intensively for the last three years. The challenges for querying data streams are unbounded memory requirement and high data rate. Thus, the computation time per data element should be less than the data rate. Also, it is very hard due to memory limitations to have an exact result. A number of techniques have been proposed to approximate the query result. One of these techniques is sliding window, in which the query result is computed over a recent time interval. Batch processing, sampling, and synopsis data structures are the other used techniques for data reduction in query processing [2].

Recently, load shedding and rate based query optimization have been proposed as a solution approach for querying data streams. Load shedding refers to the process of dropping data elements from the incoming data stream randomly or semantically in order for the query processor program to cope with the high data rate of incoming elements. Load shedding has been implemented in Auora and STREAM systems [3], [29], [30]. Rate based query optimization has been proposed to optimize the number of outputs at any specified time given that the higher the data rate the higher the output rate [31].

Mining data stream is the process of extracting interesting patterns and trends from a sequence of elements that arrive continuously in a rapid speed. Analogous to load shedding in query processing, data rate adaptation is proposed as a solution approach for mining data streams. Data rate adaptation could be used from the input side using sampling, filtering and aggregation. We propose the use of data rate adaptation from the output side using algorithm output granularity. Algorithm output granularity is the amount of mining results that fits in main memory before any incremental integration.

The AOG [13], [14] in mining data streams is unique from the other approaches by being adaptable to the data rate and the available resources of memory and time constraints. The approach has its potential by being general and could be adapted to different data mining techniques. We aim by implementing this project to provide business and scientific institutions by a real-time analysis toolset for generated data streams that became a phenomenon.

The article is organized as follows. Section 2 shows the related work in details. Section 3 discusses AOG approach in mining data streams and its formalization. The empirical results from AOG-based clustering in resource-constrained environment are discussed in section 4. Finally, the paper is concluded in section 5.

2 Related Work

This section presents a review of the state of the art in data stream mining techniques and related projects in resource constrained environments. The following subsections attempt to give an intensive overview of the current state of the field in terms of proposed mining techniques as well as applications that show the potential of the area.

2.1 Techniques

There are different algorithms proposed to tackle the high speed nature for mining data streams using different techniques. In this section, we review the state of the art of mining data streams.

Guha et al. [15] have studied clustering data streams using K-median technique. Their algorithm makes a single pass over the data and use small space. It requires O(nk) time and O(nε) space where "k" is the number of centers, "n" is the number of points and ε<1. The algorithm is not implemented, but the analysis of space and time requirements of it are studied analytically. They proved that any k-median algorithm that achieves a constant factor approximation can not achieve a better run time than O(nk). The algorithm starts by clustering a calculated size sample according to the available memory into 2k, and then at a second level, the algorithm clusters the above points for a number of samples into 2k and this process is repeated to a number of levels, and finally it clusters the 2k clusters to k clusters.

Bubcock et al. [3] have used exponential histogram (EH) data structure to enhance Guha et al. algorithm. They use the same algorithm described above, however they try to address the problem of merging clusters when the two sets off cluster centers to be merged are far apart by marinating the EH data structure. They have studied their proposed algorithm analytically.

Charikar et al [5] have proposed a k-median algorithm that overcomes the problem of increasing approximation factors in the Guha et al algorithm with the increasing in the number of levels used to result in the final solution of the divide and conquer algorithm. This techniques has been studied analytically.

Domingos et al. [8], [9], [16] have proposed a general method for scaling up machine learning algorithms. This method depends on determining an upper bound for the learner's loss as a function in number of examples in each step of the algorithm. They have applied this method to K-means clustering "VFKM" and decision tree classification "VFDT" techniques. These algorithms have been implemented and tested on synthetic data sets as well as real web data. VFDT is a decision tree learning systems based on Hoeffding trees. It splits the tree using the current best attribute taking into consideration that the number of examples used satisfies a statistical result which is "Hoeffding bound". The algorithm also deactivates the least promising leaves and drops the non-potential attributes. VFKM uses the same concept to determine the number of examples needed in each step of K-means algorithm. The VFKM runs as a sequence of K-means executions with each run uses more examples than the previous one until a calculated statistical bound is satisfied.

Challaghan et al. [6] have proposed STREAM and LOCALSEARCH algorithms for high quality data stream clustering. The STREAM algorithm starts by determining the size of the sample and then applies the LOCALSEARCH algorithm if the sample size is larger than a pre-specified equation result. This process is repeated for each data chunk. Finally, the LOCALSEARCH algorithm is applied to the cluster centers generated in the previous iterations.

Aggarwal et al. [1] have proposed a framework for clustering data steams called CluStream algorithm. The proposed technique divides the clustering process to two components. The online component stores summary statistic about the data streams and the offline one performs clustering on the summarized data according to a number of user preferences such as the time frame and the number of clusters. A number of experiments on real datasets have been conducted to prove the accuracy and efficiency of the proposed algorithm.

Keogh et al [19] have proved empirically that most cited clustering time series data streams algorithms proposed so far in the literature come out with meaningless results in subsequence clustering. They have proposed a solution approach using k-motif to choose the subsequences that the algorithm can work on.

Ganti et al. [11] have described an algorithm for model maintenance under insertion and deletion of blocks of data records. This algorithm can be applied to any incremental data mining model. They have also described a generic framework for change detection between two data sets in terms of the data mining results they induce. They formalize the above two techniques into two general algorithms: GEMM and Focus. The algorithms are not implemented, but are applied analytically to decision tree models and the frequent itemset model. GEMM algorithm accepts a class of models and an incremental model maintenance algorithm for the unrestricted window option, and outputs a model maintenance algorithm for both window-independent and window-dependent block selection sequence. FOCUS framework uses the difference between data mining models as the deviation in data sets.

Papadimitriou et al. [25] have proposed AWSOM (Arbitrary Window Stream mOdeling Method) for interesting patterns discovery from sensors. They developed a one-pass algorithm to incrementally update the patterns. Their method requires only O(log N) memory where N is the length of the sequence. They conducted experiments on real and synthetic data sets. They use wavelet coefficients as compact information representation and correlation structure detection, and then apply a linear regression model in the wavelet domain.

Giannella et al. [12] have proposed and implemented a frequent itemsets mining algorithm over data stream. They proposed to use tilted windows to calculate the frequent patterns for the most recent transactions based on the fact that people are more interested in the most recent transactions. They use an incremental algorithm to maintain the FP-stream which is a tree data structure to represent the frequent itemsets. They conducted a number of experiments to prove the algorithm efficiency. Munka and Motwani [21] have proposed and implemented an approximate frequency counts in data streams. The implemented algorithm uses all the previous historical data to calculate the frequent patterns incrementally.

Wang et al. [29] have proposed a general framework for mining concept drifting data streams. They observed that data stream mining algorithms don’t take attention to the concept drifting in the evolving data. They proposed using weighted classifier ensembles to mine data streams. The expiration of old data in their model depends on data’s distribution. They use synthetic and real life data streams to test their algorithm and compare between the single classifier and classifier ensembles. The proposed algorithm combines multiple classifiers weighted by their expected prediction accuracy. Also the selection of number of classifiers instead of using all is an option in the proposed framework without loosing the accuracy.

Ordonez [24] has proposed several improvements to k-means algorithm to cluster binary data streams. He proposed an incremental k-means algorithm. The experiments were conducted on real data sets as well as synthetic data sets. They demonstrated that the proposed algorithm outperforms the scalable k-means in most of the cases. The proposed algorithm is a one pass algorithm in O(Tkn) complexity, where T is the average transaction size, n is number of transactions and k is number of centers. The use of binary data simplifies the manipulation of categorical data and eliminates the need for data normalization. The main idea behind the proposed algorithm is that it updates the centers and cluster weights after reading a batch of transactions which equals square root of the number of transactions rather than updating them one by one.

Datar et al [7] have proposed a sketch based technique to identify the relaxed period and the average trend in a time-series data stream. The proposed methods are tested experimentally showing an acceptable accuracy for the approximation methods compared to the optimal solution. The main idea behind the proposed methods is the use of sketches as a dimensionality reduction technique.

2.2 Projects

Recent projects stimulate the need for mining data stream. These projects include:

- Bur et al. [4] have developed *Diamond Eye* for NASA and JPL. They aim by this project to enable remote systems as well as scientists to extract patterns from spatial objects in real time image streams. The success of this project will enable "a new era of exploration using highly autonomous spacecraft, rovers, and sensors" [4].
- Kargupta et al. [17], [20] have developed the first UDM system: *MobiMine*. It is a client/server PDA-based distributed data mining application for financial data streams. It should be pointed out that the mining component is located at the server side. There are different interactions between the server and PDA till the results finally displayed on the PDA screen.
- Kargupta et al. [18] have developed Vehicle Data Stream Mining System (*VEDAS*). It is a ubiquitous data mining system that allows continuous monitoring and pattern extraction from data streams generated on-board a moving vehicle. The mining component is located at the PDA.
- Tanner et al. [28] have developed EnVironment for On-Board Processing (*EVE*). The system mines data streams continuously generated from measurements of different on-board sensors. Only interesting patterns are sent to the ground stations for further analysis preserving the limited bandwidth.
- Srivastava and Stroeve [26] work in a NASA project for onboard detection of geophysical processes such as snow, ice and clouds using kernel clustering methods for data compression preserving limited bandwidth needed to send image streams to the ground centers. The kernel methods have been chosen due to its low computational complexity.

The above techniques and projects show the increasing interest in the research community in addressing the problem of mining data streams in resource constrained environments. Our AOG approach to tackle this problem is discussed in details in the next section.

3 Algorithm Output Granularity

AOG is a three-stage, resource-aware distance-based mining data streams approach. The process of mining data streams using AOG starts with a mining phase. In this step, a threshold distance measure is determined. The algorithm can have only one look at each data element. Using distance threshold in clustering has been introduced in BIRCH [33] for mining large data sets. In the mining stage, there are three variations in using this threshold according to the mining technique: a) clustering: the threshold is used to specify the minimum distance between the cluster center and the data element; b) classification: In addition of using the threshold in specifying the distance, the class label is checked. If the class label of the stored items and the new item that are similar (within the accepted distance) is the same, the weight of the stored item is increased along with the weighted average of the other attributes, otherwise the weight is decreased and the new item is ignored; c) frequent patterns: the threshold is used to determine the number of counters for the heavy hitters. A full description about AOG-based mining algorithms (LWC for clustering, LWClass for classification and LWF for the frequent patterns) could be found in [14].

The second stage in AOG-mining approach is the adaptation phase. In this phase, the threshold value is adjusted to cope with the data rate of the incoming stream, the available memory, and time constraints to fill the available memory with generated knowledge. This stage gives the uniqueness of our approach in adjusting the output rate according to the available resources of the computing device. The work has been done in the area does not pay attention to the rapid data rate of the incoming stream. The approximation algorithms and sampling techniques that have been used so far might not be sufficient with the very high data rate.

The last stage in AOG approach is the knowledge integration phase. This stage represents the merging of generated results when the memory is full. This integration allows the continuality of the mining process. Figure 1 shows the AOG-mining process.

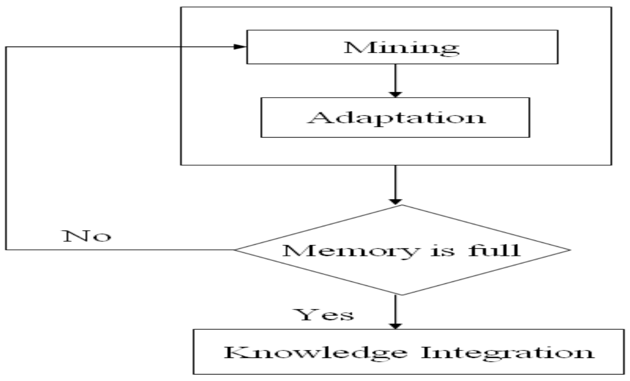

Figure 1: AOG Mining Approach

The following is a mathematical formalization of AOG-based data stream mining. Table 1 shows the symbols used in the mathematical formulation:

Symbol	**Meaning**
AAO	Atomic algorithm output size. The size of smallest element produced from the mining algorithm. For example in clustering the AAO represents the size of storing the cluster center and the weight of the cluster.
D	Duration of the time frame.
M_i	Remaining memory size by the end of time frame i. ($M_i = M_{i-1} - (AAO \times O(TF_i))$)
TF_i	Time frame i by which the threshold is adjusted to cope with the data rate.
$N(TF_i)$	Number of data elements arrived during the time frame i.
$O(TF_i)$	Number of outputs produced during the time frame i.
AR_i	The average algorithm rate during TF_i. ($O(TF_i) / D$)
DR_i	The average data rate during TF_i. ($N(TF_i) / D$)
T_i	Remaining time from the time interval threshold needed by the algorithm to fill the main memory. ($T_i = T_{i-1} - D$)
Th_i	Threshold value during the time frame i.

Table 1: AOG Symbols

The main idea behind our approach is to change the threshold value that in turn changes the algorithm rate according to three factors:

1) History of data rate to algorithm rate ratio.
2) Remaining time.
3) Remaining memory.

The target is to keep the balance between the algorithm rate and data rate from one side and the remaining time and remaining memory from the other side.

$$[(AR_{i+1} / DR_{i+1}) / (AR_i / DR_i)] = [(M_i / AR_i) / t_i] \quad (1)$$

$$AR_{i+1} = (M_i / T_i) . (Dr_{i+1}/Dr_i) \quad (2)$$

Using the Ar_{i+1} in the following equation to determine the new threshold value:

$$Th_{i+1} = [(AR_{i+1} / DR_{i+1}).th_i] / (AR_i / DR_i) \quad (3)$$

After a time frame we can use linear regression to estimate the threshold using the values obtained from the AR and Th.

$$Th = a. AR + b, \quad b= \Sigma\, th.ar / \Sigma\, ar^2, \quad a= (\Sigma\, th / \Sigma n) - (b\, \Sigma\, th / n) \quad (4)$$

Linear regression is used because of the fluctuating distribution of the incoming data elements. Data stream distribution is an effective factor in determining the algorithm output rate. The experimental results of using AOG in clustering are discussed in the following section.

4 Experimental Results

We run LWC on iPAQ with 64 MB, running Microsoft Windows CE version 3.0.9348. The program has been developed using Microsoft embedded Visual C++ 3.0. We run K-means in the same environment in order to compare the accuracy and the running time. The choice of K-means for the comparison is based on that k-means has been used in the same environment in astronomical applications due to its low complexity. Examples include: Clustering earth science data in a NASA project using K-means [27] and mission planning on-board Mars rovers using k-means [10]. In these projects, it has been pointed out that the use of k-means is due to its low complexity and the scarce of computational resources for such missions. The datasets used is the experiments are synthesized with different sizes.

The first experiment is to compare the running time of LWC and K-means with the same data set and same number of clusters generated. The following figures show the running time of LWC and K-means with different threshold values for the LWC. The number of clusters is passed to k-means to generate the same number of clusters. Figures 2, 3 and 4 show that LWC outperforms K-means even with the fine threshold that leads to creating large number of clusters.

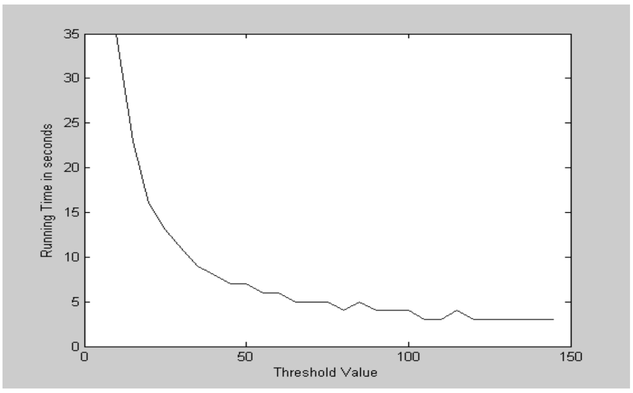

Figure 2: LWC Running Time

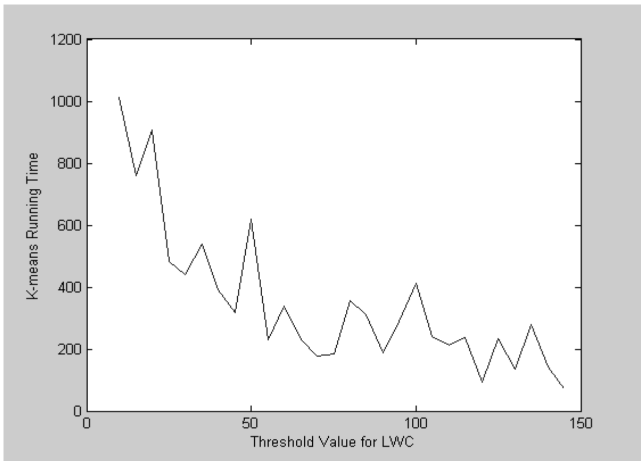

Figure 3: K-means Running Time in seconds

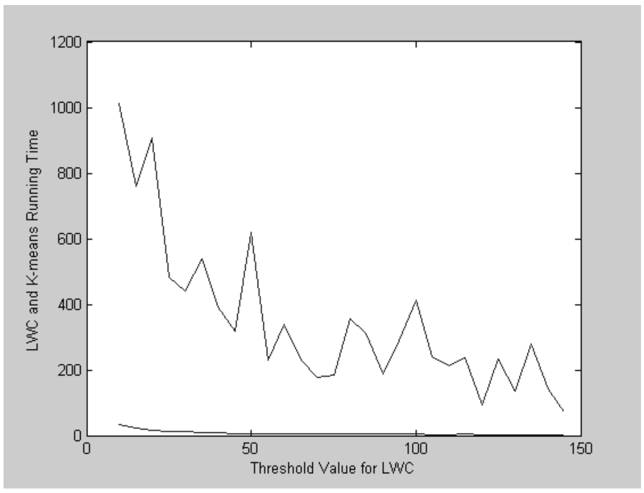

Figure 4: LWC and K-means Running Time in seconds

To show the accuracy of the LWC compared to k-means. We run the algorithms on the same data set and sorted the generated cluster centers. Figure 5 shows that the generated centers for both algorithms are very close and have the same trend.

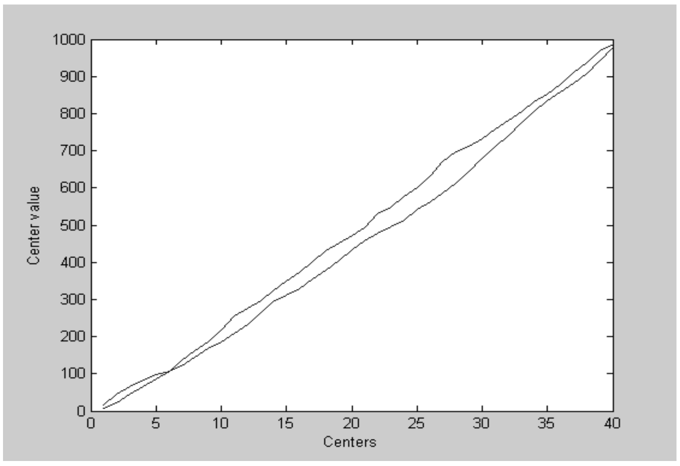

Figure 5: K-means and LWC comparison

The following experiment has been conducted to show the unpredictability of the number of passes needed by K-means. This leads to fluctuating running time with similar data set sizes. Figure 6 shows the results of this experiment.

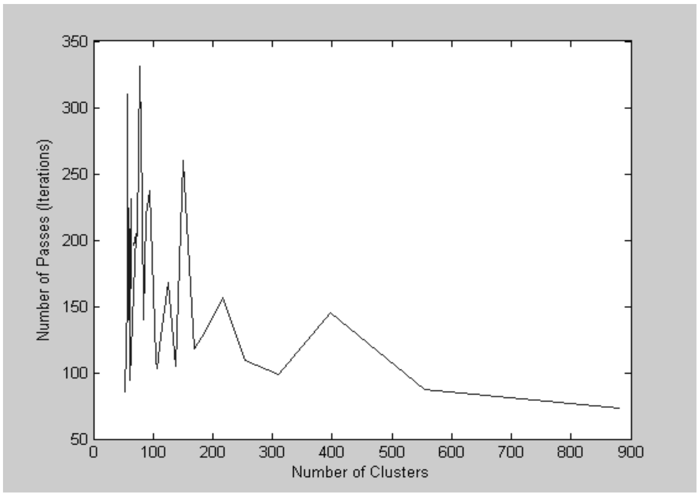

Figure 6: K-means number of passes

The above results accompanied with the results obtained in [14] show that LWC is a cost-efficient clustering algorithm that can result in an acceptable accuracy compared with traditional algorithms like K-means.

To show the scalability of AOG, we conducted this experiment. Figure 7 shows the experimental results that represent the AOG overhead. This experiment has been done with an increase in the dataset sizes. The experiment shows stability in the AOG overhead. This feature of AOG validates the feasibility of the approach with continuous streaming information.

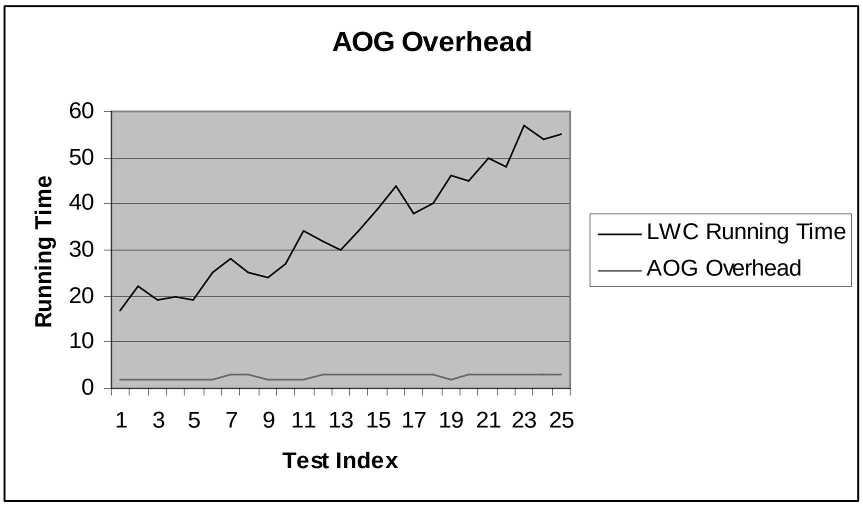

Figure 7: AOG Overhead

5 Conclusions and Future Work

The paper reviewed the state-of-the-art in mining data streams. Our AOG approach in tackling the problem has been presented with encouraging results in resource constrained environment. AOG is the first approach that pays attention to the data stream rate with respect to the available resources. The AOG-based clustering running onboard a PDA has been compared with k-means. The results showed that LWC outperforms K-means in running time with almost the same results in terms of generated cluster centers. The AOG overhead experiment shows the applicability and scalability of the approach.

- Adaptation to resource constraints.
- Generality of the approach to different mining algorithms.
- Reliability of the approach depending on AOG parameters.
- Scalability to any number of computational nodes.

The AOG-based mining package is our ultimate objective. This package has its potential by being application-independent. Different business, industrial and scientific scenarios can use the package to gain real-time insights over the generated data streams.

The adoption of AOG in querying data streams in another research future research direction. Finally, the development of AOG as a web service would enable any current streaming technique to benefit from the resource-awareness provided by the approach.

References

1. Aggarwal C., Han J., Wang J., Yu P. S.: A Framework for Clustering Evolving Data Streams. Proc. 2003 Int. Conf. on Very Large Data Bases (VLDB'03), Berlin, Germany (2003).
2. Babcock B., Babu S., Datar M., Motwani R., and Widom J.: Models and issues in data stream systems. In Proceedings of PODS (2002).
3. Babcock B., Datar M., Motwani R., O'Callaghan L.: Maintaining Variance and k-Medians over Data Stream Windows. To appear in Proceedings of the 22nd Symposium on Principles of Database Systems (PODS 2003) (2003).
4. Burl M., Fowlkes C., Roden J., Stechert A., and Mukhtar S. Diamond Eye: A distributed architecture for image data mining. In SPIE DMKD, Orlando, April (1999).
5. Charikar M., O'Callaghan L., and Panigrahy R.: Better streaming algorithms for clustering problems. In Proc. of 35th ACM Symposium on Theory of Computing (STOC) (2003).
6. O'Callaghan L., Mishra N., Meyerson A., Guha S., and Motwani R.: Streaming-data algorithms for high-quality clustering. Proceedings of IEEE International Conference on Data Engineering, March (2002).
7. Datar M., Gionis A., Indyk P., Motwani R.: Maintaining Stream Statistics over Sliding Windows (Extended Abstract). In Proceedings of 13th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2002) (2002).
8. Domingos P. and Hulten G., A General Method for Scaling Up Machine Learning Algorithms and its Application to Clustering. Proceedings of the Eighteenth International Conference on Machine Learning, 106--113, Williamstown, MA, Morgan Kaufmann. (2001)
9. Domingos P. and Hulten G. Mining High-Speed Data Streams. In Proceedings of the Association for Computing Machinery Sixth International Conference on Knowledge Discovery and Data Mining, (2000) 71—80.
10. Estlin T., Castano R., Anderson B., Gaines D., Fisher f., Judd M., Learning and Planning for Mars Rover Science, International Joint Conference on Artificial Intelligence Workshop on Issues in Designing Physical Agents for Dynamic Real-Time Environments: World modeling, planning, learning, and communicating. (IJCAI 2003). Acapulco, Mexico. August 2003.
11. Ganti V., Gehrke J., Ramakrishnan R.: Mining Data Streams under Block Evolution. SIGKDD Explorations 3(2): (2002) 1-10.
12. Giannella C., Han J., Pei J., Yan X., and Yu P.S.: Mining Frequent Patterns in Data Streams at Multiple Time Granularities. In Kargupta H., Joshi A., Sivakumar K., and Yesha Y. (eds.), Next Generation Data Mining, AAAI/MIT (2003).
13. Gaber, M, M., Krishnaswamy, S., and Zaslavsky, A., Adaptive Mining Techniques for Data Streams Using Algorithm Output Granularity, The Australasian Data Mining Workshop (AusDM 2003), Held in conjunction with the 2003 Congress on Evolutionary Computation (CEC 2003), December, Canberra, Australia.
14. Gaber, M, M., Krishnaswamy, S., and Zaslavsky, A, On-board Mining of Data Streams in Sensor Networks, Accepted as a chapter in the forthcoming

book Advanced Methods of Knowledge Discovery from Complex Data, (Eds.) Sanghamitra Badhyopadhyay, Ujjwal Maulik, Lawrence Holder and Diane Cook, Springer Verlag (2005).

15. Guha S., Mishra N., Motwani R., and O'Callaghan L.: Clustering data streams. In Proceedings of the Annual Symposium on Foundations of Computer Science. IEEE, November (2000).
16. Hulten G., Spencer L., and Domingos P.: Mining Time-Changing Data Streams. ACM SIGKDD (2001).
17. Kargupta H., CAREER: Ubiquitous Distributed Knowledge Discovery from Heterogeneous Data. NSF Information and Data Management (IDM) Workshop 2001.
18. Kargupta H., Bhargava R., Liu K., Powers M., Blair P., Klein M., Sarkar K. and Handy D.: Vehicle Data Stream Mining (VEDAS): An Experimental System for Mobile and Distributed Data Stream Mining. Information Mining for Automotive and Transportation Domain workshop. Madrid, Spain (2003).
19. Keogh E., Lin J., and Truppel W.: Clustering of Time Series Subsequences is Meaningless: Implications for Past and Future Research. In proceedings of the 3rd IEEE International Conference on Data Mining. Melbourne, FL. November (2003) 19-22.
20. Kargupta, H., Park, B., Pittie, S., Liu, L., Kushraj, D. and Sarkar, K. (2002). MobiMine: Monitoring the Stock Market from a PDA. ACM SIGKDD Explorations. January 2002. Volume 3, Issue 2. Pages 37--46. ACM Press.
21. Manku G. S. and Motwani R.: Approximate frequency counts over data streams. In Proceedings of the 28th International Conference on Very Large Data Bases, Hong Kong, China, August (2002).
22. Muthukrishnan S.: Data streams: algorithms and applications. Proceedings of the fourteenth annual ACM-SIAM symposium on discrete algorithms (2003).
23. Muthukrishnan S.: Seminar on Processing Massive Data Sets. Available Online: http://athos.rutgers.edu/%7Emuthu/stream-seminar.html (2003).
24. Ordonez C., Clustering Binary Data Streams with K-means .ACM DMKD 2003.
25. Papadimitriou S., Faloutsos C., and Brockwell A.: Adaptive, Hands-Off Stream Mining. 29th International Conference on Very Large Data Bases VLDB (2003).
26. Srivastava A. and Stroeve J.: Onboard Detection of Snow, Ice, Clouds and Other Geophysical Processes Using Kernel Methods. Proceedings of the ICML'03 workshop on Machine Learning Technologies for Autonomous Space Applications (2003).
27. Steinbach M., Tan P., Kumar V., Klooster S., Potter C., Torregrosa A., Clustering Earth Science Data: Goals, Issues and Results. KDD 2001 Workshop on Mining Scientific Dataset.
28. Tanner S., Alshayeb M., Criswell E., Iyer M., McDowell A., McEniry M., and Regner K, EVE: On-Board Process Planning and Execution. Earth Science Technology Conference, Pasadena, CA, Jun. 11 - 14, (2002).
29. Nesime Tatbul, Ugur Cetintemel, Stan Zdonik, Mitch Cherniack and Michael Stonebraker, Load Shedding in a Data Stream Manager Proceedings

of the 29th International Conference on Very Large Data Bases (VLDB), September, 2003.

30. Tatbul N., Cetintemel U., Zdonik S., Cherniack M. and Stonebraker M.: Load Shedding on Data Streams. In Proceedings of the Workshop on Management and Processing of Data Streams (MPDS 03), San Diego, CA, USA, June (2003).
31. Viglas S. D. and Naughton J.: Rate based query optimization for streaming information sources. In Proc. of SIGMOD (2002).
32. Wang H., Fan W., Yu P. and Han J.: Mining Concept-Drifting Data Streams using Ensemble Classifiers. In the 9th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD), Aug., Washington DC, USA (2003).
33. Zhang T., Ramakrishnan R., and Livny M., Birch: an efficient data clustering method for very large databases. *SIGMOD Record*, *vol.25(2), p. 103-14*, June 1996