Resource-Aware Very Fast K-Means for Ubiquitous Data Stream Mining

Rahul Shah, Shonali Krishnaswamy and Mohamed Medhat Gaber
School of Computer Science and Software Engineering, Monash University
rksha1@student.monash.edu,
{Shonali.Krishnaswamy, Mohamed.Medhat.Gaber}@infotech.monash.edu.au

Abstract. *Developments in data streams, coupled with the growth in mobile and pervasive devices, have led to the emergence of Ubiquitous Data Mining (UDM). UDM aims to perform data stream mining in a ubiquitous environment with resource-constrained and/or mobile devices. Over the past few years, stream mining techniques have attracted the attention of the data mining community. However these techniques have not addressed the problems imposed by applying the mining technique in a ubiquitous environment. Algorithm Output Granularity (AOG) has been proposed as a generic approach to enable resource-awareness in data stream mining through adaptation. AOG has been applied to lightweight mining techniques and proved its efficiency. Due to the generality of the approach, we propose to apply AOG to an efficient stream clustering technique: Very Fast K-Means (VFKM). It is an extension of K-Means for data stream clustering. VFKM is able to deal with continuous data rather than a static dataset. In this paper, we propose and develop a resource-aware version of Very Fast K-Means to enable its operation for UDM applications. Our model for Resource-Aware Very Fast K-Means (RA-VFKM) is able to adapt to variations in memory availability on mobile devices. We have experimentally demonstrated that such an adaptation enables our RA-VFKM to converge and provide results in situations (such as critically low available memory) where VFKM tends to result in an execution failure.*

1. Introduction

The customary focus in data mining has always been to develop new techniques to mine data stored in databases. However with technology growing in leaps and bounds, a number of applications in the realm of telecommunications, networking and the applications using sensors for monitoring are generating continuous flow of data known as data streams [1]. A data stream is characterised by a continuous sequence of data items arriving in a high data rate [4]. Hence, with an aim to cater for these data streams, a number of mining techniques are being developed for data streams and have led to research in data stream mining techniques [3].

Ubiquitous data stream mining (UDM) facilitates the execution of these data stream mining techniques in a ubiquitous environment. The proliferation of handheld devices, mobile phones, wearable computers and tablets [6] has enabled the growth of UDM techniques.

However, execution of applications on the said mobile devices has to cope with challenges faced in the form of constrained resources including memory, battery and CPU

processing capabilities [3]. Since these devices have limited amount of the available resources, techniques designed for execution on these resource constrained devices will have to be proficient enough to adjust and adapt depending on the resource availability. Hence extension of data stream mining techniques to a ubiquitous environment would require these techniques to be tailored to cope with the issues encountered in a resource constrained environment [3].

Algorithm Output Granularity (AOG) is a unique approach for mining data streams developed by Gaber et al. [5]. AOG enables data stream mining techniques to be resource-aware and vary its operation according to available resources. AOG is generic and has been applied by Gaber et al. [4] to lightweight mining techniques. The generality of the approach has stimulated us to apply it to an efficient data clustering technique developed by Domingos et al. [2] to prove its efficiency to enable resource-awareness to other stream mining techniques.

Very Fast K-Means (VFKM) [2] is a clustering algorithm for data streams developed as a faster version of K-Means [7], to mine large quantities of continuous rapid streaming information. Our objective is to extend VFKM to a ubiquitous environment for execution on resource constrained devices.

In this paper, we propose a novel approach to apply these concepts of resource-awareness and adaptation to VFKM using AOG. The significance of this work is that it enables VFKM to operate in ubiquitous environments.

The paper is organized as follows. Section 2 provides an overview of AOG. In section 3, we review VFKM and propose our resource-aware VFKM in Section 4. Section 5 presents the implementation and evaluation of resource-aware VFKM. Finally we conclude the paper and outline future research directions in section 6.

2. AOG: An Overview

Algorithm output granularity is a generic, resource-aware mining data stream approach that focuses on adapting the algorithm's performance according to the data rate and available memory. AOG consists of three phases as shown in Figure 1 [5]:

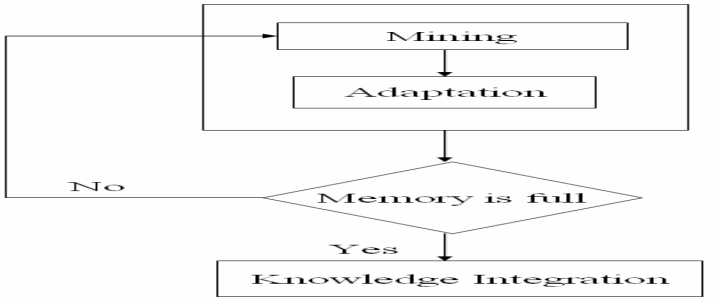

Figure 1 AOG Process

Having introduced AOG as a generic approach for enabling resource-awareness in data stream mining, VFKM is discussed in the following section as a generic data stream clustering that can benefit from the generality of AOG to enable resource-awareness in the mining process for ubiquitous applications.

3. VFKM

VFKM consists of a number of runs and each run contains a number of iterations. The number of iterations in a particular run may differ from the iterations in other runs. While KMeans uses all the data items in each of its iterations, VFKM uses only a calculated number of all the available data items. Let us assume that a database containing a large number of data items is to be mined using K-Means. Here, K-Means would tend to use all the available data items in each of its iterations. This makes K-Means inappropriate for use with data streams and highly time consuming. To overcome this, VFKM uses only a particular number of data items in each step *i*, to make the execution faster. This model enables VFKM with data streams as all the data is not available at the same time and the data arrives continuously in bursts and persistent storage of all data is not realistic or practical. VFKM observes an error $\varepsilon_{i,}$ with a probability δ_i in the results obtained by mining the limited number of data items at the end of each step *i*, as compared to the results that would have obtained by mining all the data items with K-Means in the same step. Given ε^* and δ^* which are the user defined allowable error and probability values, the goal of VFKM is to "learn in minimum time a clustering whose loss relative to infinite data is at most ε^* with probability of at least 1-δ^* [2]. Thus VFKM aims to find the cluster centroids and mine data until the termination, when $\varepsilon_i < \varepsilon^*$ and $\delta_i < \delta^*$.
The number of examples used in a particular run of K-Means is always greater than the number of examples used in the previous run. The advantage with VFKM is that VFKM's total running time is established as being less than that of K-Means.
Having discussed the two main components that form our model of RA-VFKM: AOG and VFKM, the details of our proposed approach is given in the following section followed by experimental evaluation.

4. Resource-aware VFKM

In the previous section, we reviewed VFKM and how it expedites the mining for data streams and large data sets. It is this property of VFKM that makes it a suitable candidate for UDM. In this paper, we aim to develop a resource aware extension to VFKM to enable its functioning for ubiquitous data stream mining. Resource-aware VFKM implies making the VFKM adapt to available resources.
As discussed earlier, mobile devices impose a number of constraints including memory, battery and CPU utilization. The impact of memory is significant in the processing of

learning algorithms which have substantial memory requirement in order to provide effective results in a resource constrained environment. A decrease in memory could affect the rate at which processing is carried out. Given the limited amount of memory in mobile devices, memory consumption is one of the major challenges to the execution of VFKM algorithm on incoming data streams. Hence using the AOG approach we propose to adapt the VFKM algorithm when the available free memory decreases or enters a critical stage. In the VFKM algorithm as the number of runs advance, the number of examples which are to be mined increases and this in turn necessitates additional processing and additional memory. Consequently, decrease in memory can potentially lead to an execution failure. Since the aim of our resource aware VFKM is to converge within the allowable error ε^* and probability δ^*, we propose to increase the value of ε^* and δ^* by a certain calculated factor, whenever the available free memory reaches a critical stage. Increasing the values of ε^* and δ^* means progressing more in the direction of satisfying $\varepsilon<\varepsilon^*$ or $\delta<\delta^*$ thus eventually leading to convergence of VFKM with an increased value of ε^* or δ^* than what was initially assigned. In doing so, we compromise on the accuracy by increasing the value of the allowable error. However, at the same time the algorithm converges faster when the available free memory reaches critical stage thus preventing the algorithm from total execution failure. Thus while, our strategy for increasing error and probability values compromises on the accuracy of the final results we enable convergence and avoid total execution failure in such critical situations. We now formalize resource-aware VFKM algorithm.

Stage 1: Initializations

In addition to the parameters introduced in VFKM as shown in table 1, we introduce the following parameters in resource-aware VFKM. The values of these parameters remain constant throughout the execution of the resource-aware VFKM. Table 2 shows these symbols.

Table 1 VFKM Symbols

Symbol	Meaning
k	Number of desired clusters
d	No of attributes
γ	Threshold value for a run to terminate
R_d	Range for d^{th} co-ordinate
m	Number of iterations for a run
ε^*	The value of the relative loss of mining the infinite data with VFKM instead of K-Means set before the algorithm execution.
δ^*	The value of probability for the allowable error to be ε^* set before the algorithm execution.
ε_i	Error value calculated at the end of run *i.*
δ_i	Probability value calculated at the end of run *i.*

Table 2 RA-VFKM Additional Symbols

Symbol	Meaning
ε^{**}	Maximum allowable error limit
δ^{**}	Maximum allowable probability
memory_available	The value of available free memory below which algorithm granularity is applied

Stage 2: Execution of VFKM

Once the initial parameters are set, the next step is the execution of VFKM on the incoming data stream. The execution steps of VFKM are as specified in section 3.2. However in resource aware VFKM, during every run, before checking the termination condition ($\varepsilon_i < \varepsilon^*$ or $\delta_i < \delta^*$) of VFKM, we check the available resources i.e. the memory in this case. Then depending on whether the available free memory is less than the memory_available, the values of ε^* and δ^* are modified as given in stage 3.

Stage 3: Resource-Aware Adaptation in VFKM

Before checking the termination condition of VFKM if the available free memory is found to be less than memory_available, we then increase the value of ε^* and δ^* as

$\varepsilon^* = \varepsilon^*$ + (memory used/total memory) X ($\varepsilon^{**} - \varepsilon^*$).

$\delta^* = \delta^*$ + (memory used/total memory) X ($\delta^{**} - \delta^*$)

This means that every time the available free memory goes into a critical stage the values of ε^* and δ^* (to check the termination condition of VFKM), would be increased by product of ratio of the memory used to the total memory of the device and the difference between the values of maximum allowable error and probability and the values of error and probability set before the execution of resource-aware VFKM. Thus every time memory becomes critical, the values of ε^* and δ^* would be increased (with respect to the values of ε^* and δ^* entered before the execution of resource-aware VFKM) so that there would be a greater chance of satisfying either $\varepsilon_i < \varepsilon^*$ or $\delta_i < \delta^*$ (ε_i and δ_i are the values of error and probability calculated at the end of run *i*), and the algorithm terminating after the run *i*, as compared to the normal VFKM.

The values of ε^* and δ^* set before the execution of the algorithm and the maximum permissible values ε^{**} and δ^{**} remain constant during the execution of the resource aware VFKM thus making the difference $\varepsilon^{**} - \varepsilon^*$ and $\delta^{**} - \delta^*$ constant throughout the entire course of the resource-aware VFKM.

Now, in the event of the available free memory becoming 0, i.e. memory used becomes equal to the total memory, the above equations will be transformed as follows:

$\varepsilon^* = \varepsilon^* + (1)$ X ($\varepsilon^{**} - \varepsilon^*$), and $\delta^* = \delta^* + (1)$ X ($\delta^{**} - \delta^*$)

i.e. $\varepsilon^* = \varepsilon^{**}$ and $\delta^* = \delta^{**}$

This leads to the conclusion that under the extreme condition of all the available free memory being used up, the values of ε^* and δ^* would be equal to ε^{**} and δ^{**} respectively which is their maximum permissible limits and thus would never be greater than these

permissible limits. After this stage, we go on to check the validity condition of whether $\varepsilon_i < \varepsilon^*$ or $\delta_i < \delta^*$ to determine whether or not to carry out another run.

Stage 4: Result Declaration
The algorithm terminates at the end of a particular run *i,* for which either the error calculated (ε_i) is less than the permissible error (ε^*) or the probability calculated (δ_i) is less than the permissible probability (δ^*). The centroid values of the clusters are displayed at the end on the termination of VFKM.
In this section, we have modified the VFKM algorithm to make it applicable in resource constrained devices. We include upper limits for permissible error and probability and increase the error and probability with respect to these upper limits whenever the resource constrained device gets deficient in memory. While increasing the error and probability values, we take the memory utilized into consideration. In other words, each time the available free memory in a mobile device enters a critical stage, these values are increased by a rate which depends on the amount of memory utilized.
Having presented our model for resource-aware VFKM, the following section presents experimental validation of our resource-aware adaptation to VFKM to establish its ability to demonstrate continued operation and convergence of the algorithm in a ubiquitous environment.

5. Experimental Evaluation

We conduct experimental evaluations that compare VFKM and resource-aware VFKM to establish the following:

1. Determine the convergence of resource-aware VFKM and VFKM when faced with critically low available memory.
2. Determine and study the difference in the cluster centroid results obtained by executing resource-aware VFKM against those obtained by VFKM. This allows us to establish the difference in accuracy between results obtained through resource-aware VFKM in comparison to VFKM.

We implemented VFKM on the Jeode platform which is Insignia's implementation of Personal Java 1.2. We then added the resource-aware functionality to VFKM for implementing resource-aware VFKM. We executed both VFKM and resource-aware VFKM on iPAQ H3970 with 64 MB memory, running Microsoft Windows CE. To facilitate the comparison between VFKM and our resource-aware model, we simulated data streams using a random number generator with uniform distribution and stored it so that both the programs run on the data items having the same values.
In the experiments given below, the free memory is the available free program memory in the iPAQ. To find out the number of examples to be used for the iterations in a subsequent run, we set the value of ε^* to the minimum of $\{ \varepsilon, \gamma/10\}$ to have a tighter bound.
We illustrate the experiments conducted to study the suitability of resource-aware VFKM in a ubiquitous environment. The aim of this experiment is to determine the convergence

timings of resource-aware VFKM as against VFKM on the iPAQ. VFKM and resource-aware VFKM were executed with the same amount of free program memory to start with and as explained previously, on data streams generated with the same characteristics in terms of dimensionality and range of data items values.

On execution in the iPAQ, we noticed that VFKM continued executing, completing one run after another with each run having a number of iterations. However, when the available free memory became critically low, VFKM still continued execution without making any adjustments for the decreasing memory. Effectively, after some time of continued execution with low memory, VFKM ultimately crashed due to insufficient free memory available in the iPAQ for execution. We noted the time for which VFKM executed on the iPAQ, in this experiment.

On the other hand, execution of resource-aware VFKM started similar to VFKM in the iPAQ with a series of runs. But, when the available free memory decreased, resource-aware VFKM applied the AOG to adjust as per the available memory. It increased the values of error (ε^*) and probability (δ^*), depending on the available memory eventually making the resource-aware VFKM to converge. Thus, our resource-aware approach executed a lesser number of runs as compared to VFKM, allowing VFKM to converge earlier when the memory became critical rather than continuing with the execution.

Figure 2 shows resource-aware VFKM converging due to the application of algorithm granularity in the iPAQ.

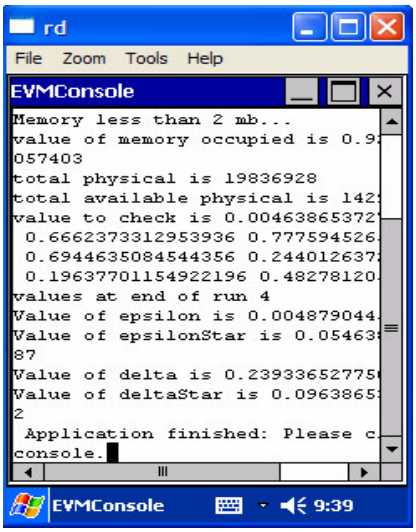

Figure 2 Snapshot of resource-aware VFKM converging on application of AOG

We conducted these experiments observing convergence between VFKM and resource-aware VFKM for eight cases, noting the time taken by both the techniques to end execution. We noticed that while resource-aware VFKM converged, the traditional

VFKM reached execution failure in all 8 times. Hence, for VFKM we noted the time for which it executed in the iPAQ before its execution failed. Table 3 lists the results obtained in each of the cases.

Table 3 Comparing Time Taken by Resource-aware VFKM and VFKM to End Execution

No	Free Program Memory (in MB) in the iPAQ	Threshold value of program memory at which the AOG is applied (memory_value in MB)	Number of iterations assumed for the first run (m)	No of desired clusters (k)	No of attributes (d)	VFKM execution failure (in minutes)	Time taken by resource-aware VFKM to converge and provide result (in minutes)
1	10.13	0.3	5	6	10	86	70
2	4.42	0.8	5	3	2	33	10
3	3.6	0.8	2	3	2	25	8
4	3.2	0.5	2	5	2	98	91
5	3.8	0.5	3	5	2	80	65
6	8	0.4	5	5	10	100	75
7	5.58	0.3	5	2	4	130	110
8	4.86	0.8	5	4	4	125	100

From table 3, it is evident that VFKM executes for a longer duration as compared to that of resource-aware VFKM. In each of the above cases, VFKM leads to an execution failure in the iPAQ due to insufficient memory whereas, the tendency to adapt with the decreasing memory makes resource-aware VFKM converge faster preventing it from an execution failure.

As outlined in the previous experiment, resource-aware VFKM reduces the execution time and also increases the allowable error in order to enable convergence. Therefore the aim of this experiment is to evaluate the loss in accuracy of resource-aware VFKM when compared to VFKM. We measure this difference in accuracy in terms of the Euclidean distance between the centroids of the clusters obtained by the 2 algorithms.

VFKM and resource-aware VFKM were executed with the same amount of free program memory to start with and in this case the generated data streams were stored and fed to both the algorithms to ensure that the exact same data values were provided to both in the same sequence. In this experiment, the difference between the cluster centroid results obtained by resource-aware VFKM and traditional VFKM are analyzed.

If the VFKM leads to an execution failure, we take into consideration the centroid results obtained in the run completed just before the failure, or else we consider the results obtained on convergence of VFKM.

We conducted seven separate experimental runs. Table 4 gives the accuracy difference i.e. the Euclidian distance between the centroids in resource-aware VFKM and those of VFKM. In other words, it estimates the inaccuracy in the results obtained by resource-aware VFKM compared to those which would have been obtained by VFKM.

Table 4 Comparing Centroid Values of Resource-aware VFKM with VFKM

No	Free Program Memory (in MB) in the iPAQ	Threshold value of program memory at which the AOG is applied (memory_value in MB)	Number of iterations assumed for the first run (m)	No of desired clusters (k)	No of attribute (d)	Euclidian distance between a centroid obtained by VFKM and that obtained by resource-aware VFKM
1	4.42	0.8	5	3	2	|c1|=0.043 |c2|=0.048 |c3|=0.055
2	3.6	0.8	2	3	2	|c1|=0.006 |c2|=0.035 |c3|=0.022
3	6	0.6	5	2	4	|c1|=0.033 |c2|=0.028
4	5.58	0.3	5	2	4	|c1|=0.036 |c2|=0.023
5	4.86	0.8	5	4	4	|c1|=0.033 |c2|=0.03 |c3|=0.018 |c4|=0.028
6	5	0.5	2	4	4	|c1|=0.029 |c2|=0.038 |c3|=0.027 |c4|=0.04
7	3.2	0.5	2	5	2	|c1|=0.03 |c2|=0.002 |c3|=0.015 |c4|=0.024 |c5|=0.035

Consider example 1 in the Table 4. Here the number of clusters is 3 with each cluster having 2 attributes. We noted the centroid of the clusters obtained by VFKM as c1 (0.66623, 0.77759); c2 (0.694463, 0.244012); c3 (0.196377, 0.482781) and those obtained by resource-aware VFKM as c1 (0.70294, 0.75509); c2 (0.6516902, 0.220971); c3 (0.196131, 0.538182). The time taken by VFKM to produce these centroid results was recorded as 28 minutes and that taken by resource-aware VFKM turned out to be 10 minutes. Now, as seen in the table above the Euclidian distance between the corresponding centroids obtained by VFKM and resource-aware VFKM for the three clusters is |c1|=0.043; |c2|=0.048; |c3|=0.055. This gives the accuracy compromised in the results by resource-aware VFKM to save 18 minutes of execution time due to a faster convergence. We notice that out of all the 7 cases listed in the table above, the maximum inaccuracy due to our resource-aware approach is 0.055 for a particular cluster centroid reading.

Thus we conclude that the resource-aware VFKM provides a faster convergence but in turn tends to compromise on accuracy. This compromise can be controlled by setting the

maximum values of permissible error (ε^{**}) and probability (δ^{**}) as per the requirement of the application. Hence resource-aware VFKM is suitable for applications which can tolerate a limited loss in accuracy (which is controlled through ε^{**} and δ^{**}).
In this section, we have evaluated our resource-aware VFKM. We can conclude that resource-aware VFKM converges faster and avoids an execution failure by adjusting to the decreasing memory. It does reduce marginally, the accuracy of the results when compared with VFKM. However, this reduction level can be controlled through our maximum allowable error parameters (ε^{**} and δ^{**}).

6. Conclusion

Ubiquitous data stream mining aims to perform the data stream mining activities in a ubiquitous environment. The key for enabling data stream mining algorithms in these environments is the ability to cope with the limited resources. In this paper, we have extended the VFKM for use in a ubiquitous environment using AOG. In our resource-aware VFKM, we move towards the convergence of VFKM, thus avoiding a total failure of the execution of VFKM when the available resources are in a critical situation. Our experiments clearly establish that while VFKM results in execution failure when resources become critically low, our resource-aware adaptation facilitates convergence and continued operation.

References

1. Babcock B., Babu S., Datar M., Motwani R. and Widom J. "Models and issues in data stream systems", ACM PODS 2002, Madison, Wisconsin, Pages: 1 – 16, 2002.
2. Domingos P. and Hulten G. "A General Method for Scaling Up Machine Learning Algorithms and its Application to Clustering", Proceedings of ICML 2001, Williamstown, MA, Morgan Kaufmann, pages 106-113, 2001.
3. Gaber M., Krishnaswamy S. and Zaslavsky A. "Ubiquitous Data Stream Mining", Current Research and Future Directions Workshop Proceedings held in conjunction with PAKDD 2004, Sydney, Australia, May 26 2004.
4. Gaber M., Krishnaswamy S. and Zaslavsky A. "Cost-Efficient Mining Techniques for Data Streams", Proceedings of DMWI2004, Dunedin, New Zealand. CRPIT, 32. Purvis, M., Ed. ACS, Pages: 109 – 114, 2004.
5. Gaber M., Zaslavsky A. and Krishnaswamy S. "Resource-Aware Knowledge Discovery in Data Streams", Proceedings of First International Workshop on Knowledge Discovery in Data Streams, held in conjunction ECML and PKDD 2004, Italy, 2004.
6. Hsu J. "Data Mining Trends and Developments: The Key data mining Technologies and Applications for the 21st Century" In the Proceedings of ISECON 2002, V 19 (San Antonio): 224b. ISSN: 1542-7382, 2002.
7. Jain A., Murty M. and Flynn P. "Data Clustering: A Review" ACM Computing Surveys, Volume 31, Issue 3, Pages: 264 – 323, September 1999.