Scaling Up for the Streaming Data

Shabia Shabir Khan, Mushtaq Ahmed Peer, S.M.K Quadri

Abstract— ***Knowledge has always been the success factor for any organization (business / technical). Survey 2012 shows that every day about 2.5 quintillion (2.5×10^{18}) bytes of data were created. As a result we are facing a challenge of handling such voluminous, potentially infinite, fast changing, temporally ordered data streams in a proper and timely manner so as to extract useful knowledge from that. However, due to its tremendous volume, we cannot store the whole of the streaming data in our limited or finite storage and due to its continuous flow we have to process it in a single pass, in contrast to the warehoused data where we could go through the data in multiple passes. In addition to this, we have to work in a limited amount of time. So, time and space are the important aspects that are taken into consideration while handling the streams of data. This paper discusses and compares those issues in the light of some sketching and counting algorithms and provides application oriented data-flow architecture for processing the streaming data along with the Granularity based approach that takes into consideration the resource awareness and adaptation for data stream mining algorithms. Further, since Analysts are mostly interested either in the recent data or in the broader view of the data, so this paper discusses a dynamic H-cube to facilitate multi-resolution analysis of streaming data wherein the Partial materialization is performed and computations are done on the fly using a tilted time frame.***

Index Terms— ***Frequency as an Interestingness Criteria, Partial Materialization, Streaming Data, Time Granularity.***

I. INTRODUCTION

When we think of aggregating data, the first thing that comes up in our mind is the use of warehouse wherein the heterogeneous data is fetched and we get an organized data in the form of facts (Fact table). The warehouse hence formed will be offline, that updates the data periodically or online that is active all the time to check for new updates in the events that have been recorded previously [21]. Further we can get the analytical reports from it, or we provide the warehoused data to any mining algorithm or visualization tool. The subjective data needs to be warehoused i.e. Store and aggregate facts on the basis of the type of query that is being asked , however , in case of voluminous data , it becomes quite hectic to warehouse the data each time a query is sent. "When I see a thick novel, I just read its summary usually written at the end of the cover page and I get the idea regarding the information, it really wants to deliver", same goes here! We shall go for summarization or aggregation of the data. In addition to this, it's really inappropriate to spend lot of time in scanning, filtering and storing the data, which seems to be of no use when we have to answer a particular query such as "which is the most frequent item?" We need not to spend time and memory (because we may run out of memory) in handling the least frequent items as well. Warehousing the streaming data is really impractical here and we just need "On-the-fly" analysis. If the data is appropriate for our query, then we can use the space for it; if not, then reject or remove the data. The streaming techniques are based on this principle. Several Models/ Techniques have been presented to analyze the voluminous data:

A. *Sampling the data stream randomly:*

As it is impossible to analyze the whole stream and know in advance the size of the entire stream, we try to obtain the unbiased random samples of data at periodic intervals. The method of Sampling also helps us in reducing the running time of the computations as we are focusing on a much smaller sample. In the Reservoir sampling technique, a reservoir having minimum size of say 'r' is maintained, wherein the size of reservoir is directly proportional to its cost. From this reservoir, random sample of size 'r' is generated on which the Analysis operation is being done, rather than on the whole stream. If there is 's' number of elements seen so far, then the probability of the new arrived data element for replacing any previously sampled element is (r/s). Being a fundamental technique for analyzing data stream it proves to be beneficial; however it has the risk of leading to inaccurate information In fact, it all depends on the type of sequential pattern the data is having, which is again difficult to analyze in case of streaming data .[4],[5]

B. *Recent data analysis:*

This is possible through the sliding window that tries to concentrate on the most recent elements in the data stream and ignoring the rest of the elements. So, here the decisions are made on the recent data. Considering the size of window as 's', i.e. the number of the elements selected within the window is 's' and time 't' being the time of arrival of a new element, then the new arrived element will expire at time (s+t). The memory requirement has been reduced by this technique; however the decision is being taken on the basis of the most recent data that can prove useless for those situations wherein the whole of the data can lead to an extremely different and appropriate decision.

C. *Sampling from the Sliding Window:*

Extracting samples from such a slide is of no importance because of the implicit nature of the expiration of elements in the window. This can be resolved by the technique of oversampling wherein a sample size > s is kept reserved for avoiding the s- sample size to be expired whenever sample size of 's' is required; however the increase in cost and non deterministic bounds of memory can finally lead to the decrease in the performance.

Manuscript published on 30 December 2012.
***** Correspondence Author (s)

Shabia Shabir Khan*, Computer Science Department, University of Kashmir, Srinagar, India
Dr. Mushtaq Ahmed Peer, Computer Science Department, University of Kashmir, Srinagar, India
Dr. S.M.K.Quadri, Computer Science Department, University of Kashmir, Srinagar, India

Further, using the fixed size or Bursty window, the sampling with or without replacement is an improvement to the oversampling technique. The arrival of elements in such a fixed sized window is 'one at a time' wherein the most recent 'n' elements are of concern. In contrast, Bursty Window accept the arrival of elements in 'bursts' wherein the elements from last 'k' steps are of concern [4],[5],[6],[7].This technique has proved useful but each time the focus is on the small portion of data that cannot work in situation where we need to analyze the data at a time as a whole.

D. Frequency distribution:

One of the synopsis data structures for the frequency distribution of elements in stream is Histogram approach. It is an attempt to build synopsis of large data streams. Here, Instead of sampling, the technique tries to provide the approximate answers to the queries. However the range of each bucket used for partitioning the data set is still a major concern. Formally, to overcome it, we can just keep the size of each bucket same, but the better advancement is the use of V-Optimal Histogram wherein the focus is on the minimization of the frequency variance within each bucket and further minimize the squared error. In such an improved technique we suppose a value, say 'h_r', being the histogram estimator for all the element values in a particular bucket or we can say it is a single point into which sequence of element values in a particular bucket, say 'r', collapse. Each element in the list or bucket , say ('a_r', 'b_r',….,'e_r'), having individual frequencies, say ('f_{ar}','f_{br}',…'f_{er}'), respectively, is considered to have 'h_r' only as the estimated frequency, based on histogram.

Then for each element k , in bucket r ;

Error for 'k' = Estimated frequency for the bucket 'r' - Actual frequency of element 'k'

$$= h_r - f_{kr}$$

Squared Error for the values in bucket 'r'=Total Distance between Histogram Val. and Actual Val.

$$=\sum_{k = ar, br,\ldots,er} (h_r - f_{kr})2$$

So, the main focus of V-Optimal Histogram is on the minimization of the Squared Error because the smaller the Squared Error, the better is the approximation. However, histograms require multiple passes over the data.

E. Using Hierarchical Structure to provide various levels of Resolution (Wavelet Transform):

The main aim behind using a multi-resolution data structure is to provide detailed understanding of data stream at multiple levels of resolution. Multilevel actually indicates the various levels of abstraction of data (e.g. hour-wise -> minute-wise -> second-wise). The structure thus formed is balanced binary tree-like structure wherein the balance is maintained each time a new element of the data stream arrives. Each level represents the level of resolution. This technique is sometimes called Multi-resolution Analysis (MRA). Such technique is an approximation to histogram for query optimization.

The concept of clustering can be used to form the various levels of resolution. The greater the distance from the root level, the more is the detailed level of resolution. This 'Divide and conquer' method of Multi-resolution can balance the two opposing situations or qualities i.e. Accurateness and Storage Space. Wavelet transform, a signal processing technique uses same clustering approach which decomposes an input signal into different frequency sub-band by building a hierarchy structure over streaming input signal. Such technique is used for representing data in reduced or compressed form. Wavelet transform technique is a lossy technique wherein we reconstruct approximate of the original data after the dimensionality reduction, in contrast to lossless technique, wherein the reconstruction produces the original data back. Such a technique applied on data vector of 'C' as coefficient, produces its strongest approximated coefficient as 'C' depending upon the threshold that filters out the strong ones.

Scaling in wavelet transform of a signal to be processed:-

Low scale -> compressed wavelet -> Rapidly Changing details -> High frequency

High Scale -> Stretched wavelet -> slowly changing details -> Low frequency

At lower frequencies we use probably the less number of samples, thus, as the frequency decreases, the time resolution also decreases and at such lower frequency, with the decrease in the frequency interval, the frequency resolution increases. Some applications of such Wavelet Transform technique are found in Compressing images, Identifying pure frequencies, removing noise from signals, Detecting discontinuities, etc. A key advantage of wavelet transforms is that it goes for temporal resolution in addition to frequency resolution as well and thus has proved very useful for analyzing time series data in addition to other types of spatial and multimedia data. Wavelets are efficient in handling Surprise Queries such as, "what is the sudden change in the temperature of a particular state in a particular month", in contrast to the trend analysis queries such as "how the temperature is varying in a particular year". Under such trend query, we just record the trend for once and all and easily answer any query related to temperature at any particular month or even day. However wavelets are specially used in case a sudden change occurs that is out of the recorded trend. The single issue that is in such hierarchical structure is that the termination condition or the threshold for decomposition has to be specified and in fact require multiple passes over the data. [4], [8], [9], [10]

II. SINGLE PASS TECHNIQUE

Sampling has proved to be really unsuitable technique if we ever want to count distinct items in the stream. If a large fraction of items are sampled, we exactly don't know if they are all same or all different or contain just some of the distinct elements. In addition to this, we need a technique that helps in overcoming some of the issues, like multiple passing through the fast data streams of data.

As we know the frequency moment 'F_k' of a data set represent important demographic information about the data, and is defined as the sum of k-powers of the numbers 'f_i' (frequency of i^{th} element),thus provide useful statistics on the sequence. Actually, 'F_K' indicates the degree of skew of the data, which is of major consideration in many parallel database applications such as in the determination of the selection of algorithms for data partitioning or query answering. Let us assume a data stream of objects or elements $S = \{q_1, q_2, q_3 \ldots\ldots q_n\}$ wherein q_i is element of O; O= $\{O_1, O_2, O_3,\ldots\ldots,O_m\}$. i.e., for each object 'i' in Universe or domain, we want to maintain the frequency 'f_i'or number of occurrences of 'i' in the sequence 'S'.

Let's take into consideration the frequency moments of S. The kth frequency moment F_k of S for real $k > 0$ where 'm' is

$$F_k = \sum_{i=1}^{m} f_i^{\,k}$$

the universe or domain size and 'f_i'
is the frequency of 'i' in the sequence. In particular, we represent:

'F_0' as the number of distinct elements appearing in the sequence,

'F_1' as the length of the sequence (i.e., 'n' here) &

'F_2' as the self-join size / Gini's index of homogeneity /repeat rate representing the surprise index of the sequence.

F_k for $k \geq 2$ can indicate the amount of skew (representing the distance of the frequency distribution of items from the uniform state) of a data stream. If 'O' is large, then, this structure can get quite large as well. However, when the amount of memory available proves to be smaller than 'm' ,we obviously would like to have a smaller representation instead. So, we use a specific kind of synopsis (approximation method) known as Sketches representing an estimation of the frequency moments that can work in a single pass and make use of both small space and small time. Sketches are useful for estimating frequency moments [4], [11]. It is for the sake of efficiency, that the computation of estimates for frequency moments in one pass under memory constraints is being done .This is performed by building a small-space summary for a distribution vector (e.g., histogram) using randomized linear projections of the underlying data vectors. Given the 'n' elements in a sequence and 'm' elements in the universe set of distinct elements , Sketches provide an approximation of 'F_0','F_1','F_2' in space complexity of O(log m +log n). The basic working behind this is to hash every element uniformly at random to -1 or +1.Further we compute a random variable defined by the formula;

$$X = \sum_i f_i r_i,$$

Where 'r_i' is element of {-1,+1}, thus indicating the displacement of the item from the midpoint. By squaring this displacement i.e., X^2, we estimate the data skew, F_2, the surprise index of the sequence [4].

An important sketch based method is Count-Min Sketch. The counter based method is the heavy Hitters method

III. COUNT-MIN SKETCH

This method provides estimation of frequency-related properties e.g. Estimating the frequencies of items or finding out the most frequent item taking into consideration the memory efficiency. If the stream of data contains set of values with duplicates, we need to find out the number of duplicates or frequency for each distinct element. The basic idea behind Count-Min Sketch is somehow similar to Linear Counting wherein the sketch is simply a two-dimensional array (d x w) of integer counters i.e. total of (d*w) counters .Each incoming element is mapped to one position at each 'd' rows by simply hashing it through ' d' different hash functions, thus maintaining a dedicated counter for each value using a hash table. The value of d is equal to (log 1/δ) wherein 'δ' is user specified. All the unseen elements have will be initialized to zero. Each occurrence of the value in the sequence increases its corresponding counter. However, the frequency estimation in Count-Min algorithm takes into consideration the minimum of the corresponding counters because of the estimation error 'E' that is always positive due to the collisions that can cause additional increment to the counter in case of streaming data [4],[12],[13],[16]. It has been analyzed that the space complexity is $O(1/_E \log 1/_\delta)$ and Probability of more error is less than (1- δ).

The following statements can revive the idea about Count Min sketch.

```
For each element 'ITEM' in the stream
    For each hash function $hash_i$ (ITEM)
        Position = $hash_i$ (ITEM)
        Update $Array_i$[W] + 1
```

However, the major drawback in such a sketching algorithm is the lack of accuracy.

IV. HEAVY HITTERS METHOD

It is a counter-based Stream-Summary technique that allows one to detect most frequent items in the dataset and estimate their frequencies with explicitly tracked maximum potential error of the estimation [17]. It traces a fixed number of more frequent elements and if one of these elements occurs in the stream, the corresponding counter is increased. On the arrival of a new non-traced element, the least frequent traced elements replaced by it and the element that is replaced becomes non-traced.

The procedure for estimating the frequency-related attributes can be illustrated with an example:
Example:

Consider an input stream {a, a, b, b, c, a, d}, the traced 3-fixed number of frequent elements {a,b,c} , the bucket '$FBCK_i$' corresponding to the particular frequency 'i', where $i \geq 1$ and 'PER' as the potential error. We start with having no bucket for frequency .On the arrival of the stream element 'a', a new $FBCK_1$ is created wherein the frequency of 'a' is 1. Here the value of 'PER' remains zero. Further next again 'a' arrives, so the respective slot of 'a' is being removed from the $FBCK_1$ and attached to $FBCK_2$ having frequency of 2. $FBCK_1$ is deleted because it is now empty. Now 'b' arrives. Since this is its first occurrence, so it is being attached to the $FBCK_1$ only. Then as the next element b arrives again, the respective slot of 'b' is removed from the $FBCK_1$ and attached to $FBCK_2$. $FBCK_1$ is again deleted because no element is attached to it .Further, as the next element 'c' arrives; it is attached to $FBCK_1$ simply. Next arrives element'd'. Being a non-traced element it throws out and replaces element 'c' in $FBCK_1$ because 'c' in $FBCK_1$ was having the lowest frequency. The number of occurrences of the element 'c' with frequency of 1 is considered to be the 'PER' for the element'd'. So, in this procedure, we need to scan elements in the high frequency buckets.

A. Lossy counting algorithm :

Keeping track of a large majority is really not practical when it comes to streaming data. There should be a method in which we could periodically delete the less frequent occurrences i.e. having very less count .The Lossy counting algorithm works by grouping the data items into blocks and follows the procedure of counting within each block. [4],[17].This algorithm is very simple and efficient as well. The procedure is to some extend similar to Apriori algorithm, that also looks for interestingness criteria e.g. frequency of the elements.

Here, in the similar way , we have a threshold or minimum support say, 'M' ,for checking whether the element is eligible to be considered as frequent or not. In addition to this we have a user defined acceptable limit of error (acceptable scale of the error), say, 'E' .It's the user who can decide how big the error can be.

Following points need to be taken into consideration:

- The whole N-Sized stream is divided into S-Sized blocks 'B' of items.
- Under each step 'i' of the algorithm , the possible error 'C_e' on counted frequency 'F' will be (B_i –1), wherein B_i is the block number in the ith step and ($B_i \leq N/S$)
- The basic principle of working lies under the following condition:

```
For each element in the block $B_i$
    {
    If  $(F + C_e) > B_i$, then Accept the item to be among the
            most  frequent ones
    Else,  Delete the item;
    }
```

- The user-specified parameters are minimum support threshold 'M' and error parameter 'E' that have value between the range (0,1) . The minimum support 'M' of whole stream seen so far, i.e. (M*N) will be the actual frequency that is considered to be acceptable for including the corresponding item among the most frequent ones.
- However, the technique is an approximation, wherein some degrees below the minimum support (M) should be accepted . The degree up to which it is accepted is 'E'-the acceptable scale of error.
- The value of 'E' is usually computed as one-tenth or one-twentieth of the minimum support (M). e.g. For minimum threshold of 1.0% we get 0.1% as the acceptable error E.
- We now compute the new acceptable frequency as (M-E)*N, and the value of (E*N) is the frequency that will not all be accepted.
- All individual frequencies are usually less than their actual frequencies, depending on the user specified parameters.

Example: We start with the user specified parameter E =0.2, which is an acceptable scale of error.

Since we have, W = 1/E , so , we can easily calculate the size of each block. So here the number of items in each block be 5 i.e. 'W'=5.

We have the stream as follows:

AADCDCDBDHHGCHB AHBDG

Dividing the whole stream into 4 blocks of size 5:

AADCD, CDBDH, HGCHB, AHBDG

After Applying Lossy Counting Algorithm, we saw that output list contains the Item D with Frequency =5, and Item H with Frequency =3. The actual acceptable frequency in the output list for this example is 5, however we could see that the technique accepted the item H as well, showing in the last its frequency as 3, although we can see that the in actual item H was having the frequency of 5.But being rejected in the first 2 steps really did not affect its acceptability. This is due to the extra error degrees that we consider as acceptable. The random arrival distribution in the streaming data is solved easily by this technique. Analysis has shown that at most entries in computation for random streams by this technique bounds to ($1/_E$ log E*N) while as (7/E) for those data streams wherein the low frequency elements occur more or less uniformly at random [4],[15].

B. Probabilistic Lossy counting algorithm:

The Lossy Counting algorithm is used for finding the approximate frequency counts in the data streams and has been the most efficient for finding heavy hitters. However , Analysis has shown that the memory consumption exhibited by lossy Counting Algorithm is more than a technique called Probabilistic Lossy Counting , which is its advanced version [18].The difference in the two versions is due to the possible error 'C_e' on counted frequency 'F' used in lossy Counting Algorithm. As the block number increases , the calculated value of C_e makes the set of elements remain in the output list for a longer period of time ,unless and until the removing condition [$(F + C_e) \leq B_i$] is satisfied , thus consuming more space. In contrast the Probabilistic Lossy Counting Algorithm has improved by assigning a smaller value to C_e such that the set of elements remain in the table only for a shorter period , thus providing a full on efficient algorithm. Probabilistic Lossy Counting has the same memory bounds i.e ($1/_E$ log E*N) and (7/E) for those data streams wherein the low frequency elements occur more or less uniformly at random[39]. As we know the possible error bound parameter is used to remove the non-frequent elements and has direct impact on the memory consumption of the algorithm. However for Probabilistic Lossy Counting Algorithm this bound P_e is made comparatively smaller due to which the elements temporarily in the output list shall stay for shorter period of time, thus lowering the memory consumption and further guaranteeing that the error on the frequency of an element is smaller than the bound P_e with a desired probability of ($1 – \delta$) where $\delta \ll 1$.

The Probabilistic error bound is given as: [17]

$P_e = (\delta (1 - (B_i-1)^{\beta}) + (B_i-1)^{\beta})^{1/\beta}$ Or,

$P_e = \beta$th root of ($\delta (1 - (B_i-1)^{\beta}) + (B_i-1)^{\beta}$)

Where,

B_i is the window at ith step

δ is some small probability, say 0.05

β is the power-law distribution parameter..

The methodology of Probabilistic Lossy Counting Algorithm is quite similar to its previous version, except the application of new error bound, and provides a better, similar, or only slightly worse performance than the lossy counting Algorithm. Going through this algorithm our goal was same i.e. To have the algorithm that uses as little memory as possible. The overall Flow that take place in Lossy Counting Algorithm is shown in figure1 below:

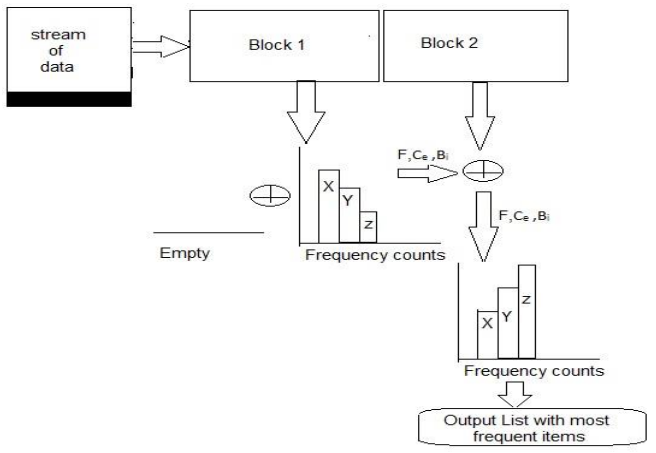

Figure 1:The Process of filtering out frequent Items using blocks of data stream.

V. FREQUENT ITEM SETS

It is obvious that, the more the number of frequent items, the more number of frequent item-sets will be created. So, following the procedure of flow processing, block by block, would really result in the ‘out-of-memory’ condition. So we go for the processing of number of blocks in one go. A number of blocks shall be enclosed within a outer block, say a ‘Frame‘. We start by dividing the incoming transaction stream into blocks of size W = 1/E , i.e. block consisting of ‘w’ transactions and reading as many as possible into the main memory .In order to read such batches of transactions into available main memory repeatedly is done by buffer . Suppose ‘α’ be number of blocks in the current frame that is being processed .Each frame can have up to 30 blocks.
As each frame of ‘α’ blocks arrives, we count the occurrences of the item set and update the corresponding frequency. We remove the entry if the condition $[(F_i + C_e) > B_i]$ is not satisfied where B_i is the current block number. An item set is inserted as a new entry in the output list if it has the frequency $F_i \geq \alpha$, but does not appear in the output list. For this entry, C_e is set to $(B_i - \alpha)$. By this technique, the item sets with frequency $Fi < \alpha$ would not be listed anymore and thus plays an important role in saving the memory. The smaller the value of α, the more are the false subsets in the output list, thus affecting the efficiency of the algorithm in terms of both time and space.

The three important Structures that help us in performing the filtration of most frequent item sets are:

- The buffer repeatedly reads the frames of transactions into the memory and sorts them on the basis of item-id.
- Trie maintains the structure S wherein the entries are of the form {item-id, F, S, level} .
- Set Generator operates on current batch of transactions by creating / removing an entry from the output list depending upon the condition set for the algorithm. The sets are produced are lexicographic [4]

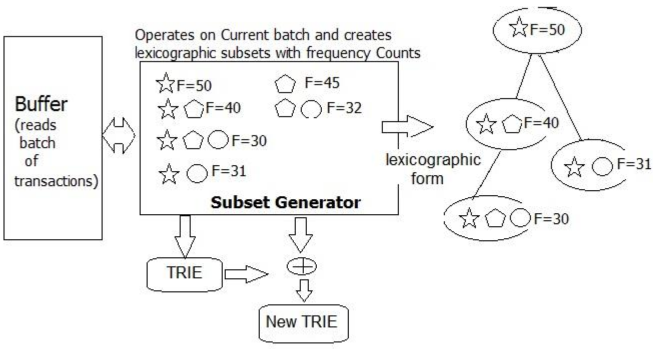

Figure 2: The Process of filtering out frequent Item Sets using lexicigraphic form

Further, Association rule mining is an important tool for knowledge discovery and is applicable to other data mining tasks such as clustering and classification. The most frequent items sets that have been already filtered can be further passed to the process of Association wherein an important threshold called confidence provides us interesting patterns or rules that can be of great help to the analysts.

VI. RESOURCE CONSTRAINT-SOLUTION FOR THE SCARCITY OF RESOURCES

However we have an approach of the granularity that controls the consumption of resources over a period of time (time frame) on the basis of resource availability (RAM, CPU, battery etc.)As in the Algorithm of finding the frequent item sets, explained above in section IV and V , the collection of the of frequent subsets in the frequency list are enclosed within a time frame wherein the input data stream arrival rates are changed that cope with the availability of the data resources, the output size (knowledge structure or hidden patterns) is changed in an attempt to preserve the limited memory space and in addition to this , changing the algorithm parameters and thus reducing the processing power[19],[22]. The three different variations in the Granularity approach are:

- AIG (Algorithm Input): It is the process by which the input data rates are adapted on the basis of battery charge.
- AOG (Algorithm Output) :It is the process by which the output data rates are adapted on the basis of memory.
- APG (Algorithm Processing) It is the process by which the processing settings are adapted on the basis of the CPU usage.

There is a collective collaboration among the algorithm input, output and processing settings. The implementation of the granularity approach on the data mining technique primarily requires the need for keeping track of the computational resources over fixed time frames. By such periodic monitoring, the mining algorithm changes its parameters related to input, output or processing. Also the adaptation overhead is inversely proportional to the time interval parameter that we set .i.e. the more the time interval, the less will be the adaptation overhead. However the time frame interval is directly proportional to the resource consumption risk .i.e. the long the time interval, the more will be the risk of high consumption. The basic principal of working is based on the following statements for each resource ‘r’ :

$$if\ \left(\frac{\text{Time left to last the Total Application LifeTime}}{\text{Time Frame Interval}} > No.of\ frames\ to\ consume\ a\ Resource\ 'r'\right)$$

Then it indicates Lower Consumption of resource r,
Else, indicates higher consumption of the resource r;

Accordingly the resource awareness can help us to a great extend to the resource constrained streaming data mining.

VII. RECENCY OF THE DATA

The data obtained from such frequency counting algorithms may comprise of the recent as well as the historical. In addition to this, analysts always show interest in the recent data at finest level (e.g. quarter- or hour- wise) but long term changes at coarser level (e.g. month- or year- wise). To specially emphasize the recency of the data, that is important for analyzers, time granularities are integrated with the procedure of frequency counting technique. This idea is supported by the application of Stream Cubes [20]

Stream Cube: For efficient and effective computation of stream cubes we have the three steps to follow:

a) Firstly, We shall use the concept of tilted time frame by adopting one of the three kinds of multi-resolution models to register time-related data:-

- Model wherein the granularity is based on the natural time scale: e.g. second, minute, quarter, hour, day, week, months.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)
© Copyright: All rights reserved.

- Model wherein the granularity is based on the Logarithmic time scale: e.g. 4 quarters, 8 quarters, 16 quarters, etc
- Model wherein the snapshots held in each frame are stored at differing levels of granularity , the frame number varying from 1 to N satisfying the condition :

$\log_2(T) - C \leq N \leq \log_2(T)$ with 'T' being the Clock time elapsed since the beginning of the stream, 'N' being the maximum number of frames and 'C' the Maximum number of snapshots in each frame.

b) Next, what is really needed is a realistic arrangement to compute on-the-fly and build dynamic cuboids out of the whole cube. This is because the data is voluminous but we have limited memory space .So we propose to maintain a small number of critical layers. The two important critical cuboidal layers that are used to online analyze the subjective data stream are: -The minimal interest layer (m-layer), wherein the data is aggregated along the popular path towards the next layer i.e. observation layer (o-layer). Applying the tilted time frame granularity, the registration of the most recent time is done at the finest granularity level and the more distant time is registered at coarser granularity level.

c) Thirdly, a procedure (algorithm) is required to perform efficient computations. Materialization is important because non materialization results in slow query response time. However, full materialization can consume more space and time. Thus we go for the partial materialization with a tilted time frame as its time dimension, where only those cuboids are materialized that come along the popular path between these two critical layers. So, in order to avoid the unrealistic demand on space and time and perform flexible analysis, whenever a query request or a drill-down clicking event arises that requires to compute the cells, which are slightly outside the popular path, we just find the closest lower level computed cells and use intermediate computation results to compute the measures that are requested. This is possible due to the corresponding linked list nodes that contribute to the cell. For setting up partially materialized data cubes and for the incremental and online cubing of streaming data, we need an efficient data structure , which is known as H-tree /Cube that represents a popular path in the corresponding data cube. [2], [3][20].

VIII. OVERALL FLOW ARCHITECTURE FOR PROCESSING STREAMING DATA

The journey of the streaming data from the operational systems towards the knowledge extraction is shown in the data flow architecture below. The whole of the data is passed through the ETL/ELT (extract, transform, and load) [21] and accordingly forwarded to the stream engine, in contrast to the static data that can be warehoused easily. Further, from streaming engine, the reduced data (sampled or filtered frequent item sets etc.) are forwarded to the Granularity based Technique, wherein the mining process (Association pattern mining etc.) is performed taking into the consideration the resource availability awareness. As soon as the resource seems to be unavailable, the mining process is stopped and the knowledge patterns extracted till that point are forwarded to the Analysts or End users. The overall application oriented data flow architecture is presented in figure 3 below.

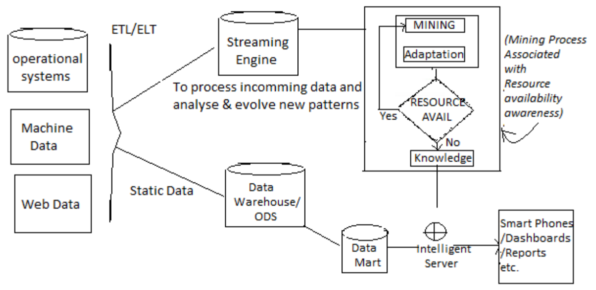

Figure 3: Data Flow Architecture for Streaming data with resource Constraints

IX. CONCLUSION

In this paper we , we discussed the various stream based algorithms that compute approximate counts of elements in a data stream and thus relieve us from processing the whole of the stream. The approximation of the data is done within the error factor 'E' of the actual answer. We took into consideration the most important interestingness criteria and that was frequency. The problem for identifying frequent item sets using a probabilistic Counting algorithm produces approximate results with less consumption of memory. We have extended this algorithm by applying the concept of Granularity on the Adaptation basis that takes into consideration the resource constraints. The recency of the data is gained by further processing the data on the basis of time granularity using tilted frame. In the future, we plan on adding additional functionality to stream especially in the exploration nuggets of knowledge.

REFERENCES

[1] Nan Jiang and Le Gruenwald,"Research issues in Data Stream Association Rule Mining",*SIGMOD* Record, Vol. 35, No. 1, Mar. 2006.

[2] J. Han, J. Pei, G. Dong, and K. Wang,"Efficient computation of iceberg cubes with complex measures" *SIGMOD*, 2001.

[3] J. Han, J. Pei, and Y. Yin.," Mining frequent patterns without candidate generation", *SIGMOD*, 2000.

[4] Jiawei Han, MichelineKamber , Book:"Data Mining: Concepts and Techniques";*Morgan Kaufmann Publishers*

[5] Vladimir Braverman, Rafail Ostrovsky, Carlo Zaniolo ," Optimal sampling from sliding windows", *PODS '09 Proceedings of the twenty-eighth ACM SIGMOD-SIGACT-SIGART* symposium on Principles of database systems , Pages 147-156

[6] A. Arasu, B. Babcock, S. Babu, J. Cieslewicz, M. Datar, K. Ito, R. Motwani, U. Srivastava, J. Widom, "STREAM: The Stanford Data Stream Management System," Book Chapter –"Data-Stream Management: Processing High-Speed Data Streams", *Springer-Verlag*, 2005.

[7] Graham Cormode , S. Muthukrishnan , Irina Rozenbaum, "Summarizing and mining inverse distributions on data streams via dynamic inverse sampling", *Proceedings of the 31st international conference on Very large data bases*, August 30-September 02, 2005, Trondheim, Norway

[8] Guy P. Nason and Rainar Won Sachs, Phil. Trans. R.Soc.Lond,"A Wavelets in time series an analysis ", *IEEE Int'l Frequency control Symposium* (2000)

[9] R.J.E. Merry ,"Wavelet Theory and Applications, A literature study", DCT 2005.53Eindhoven, June 7, 2005 ,

[10] Mohammed A. H. Lubbad and Wesam M. Ashour, "Cosine-Based Clustering Algorithm Approach ", Copyright © 2012 MECS , *I.J. Intelligent Systems and Applications*, 2012, 1, 53-63

[11] Noga Alon, Yossi Matias, Mario Szegedy, February 22, 2002, "The space complexity of approximating the frequency moments ",A preliminary version of this paper appeared in Proceedings of the 28th *Annual ACM Symposium* on Theory of Computing (STOC), May, 1996.

[12] Lukasz Golab and Theodore Johnson, "Consistency in a stream Wraehouse",5th Biennial Conference on Innovative Data Systems Research(*CIDR'11*),California , USA

[13] F. Deng, D. Rafiei,"New Estimation Algorithms for Streaming Data: Count-min Can Do More", 2007, http://webdocs.cs.ualberta.ca/

[14] Gurmeet Singh Manku, Rajeev Motwani, "Approximate Frequency Counts over Data Streams " , *Proceedings of the 28th VLDB Conference,* Hong Kong, China, 2002

[15] Graham Cormode and S. Muthukrishnan ,"An Improved Data Stream Summary:The Count-Min Sketch and its Applications", preprint submitted to Elsevier Science 16 December 2003

[16] A. Metwally, D. Agrawal, A.E. Abbadi, " Efficient Computation of Frequent and Top-K Elements in Data Streams", In*: International Conference on Database Theory* (2005).

[17] Xenofontas Dimitropolos, Paul Hurley, Andreas Kind, "Probabilistic Lossy Counting:An efficient algorithm for finding heavy hitters", *ACM SIGCOMM* Computer Communication Review 7 Volume 38, Number 1, January 2008.

[18] Mohamed Medhat Gaber , Shonali Krishnaswamy , Arkady Zaslavsky, "Resource-aware Mining of Data Streams" , *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

[19] Ajith Abraham, Aboul-Ella Hassanien, André Ponce de Leon F. de Carvalho, Vaclav Snášel, "Foundations of Computational Intelligence: Volume 6: Data Mining" , *Springer* 2010

[20] Jiawei Han, Yixin Chen, Guozhu Dong,Jian Pei, Benjamin W.Wah, Jianyong Wang,Y,Dora Cai, "Stream Cube: An Architecture for Multi-Dimensional Analysis of Data Streams", Distributed and Parallel Databases, 2005, *Springer Science + Business Media, Inc.* Published online: 20 September 2005

[21] Shabia Shabir , Dr. Mushtaq Ahmed Peer ,"Expedition for the exploration of Apposite Knowledge" , *IJCSIT-2012(IJCSIT) International Journal of Computer Science and Information Technologies,* Vol. 3 (5) , 2012,5164 – 5168.

[22] Mohamed Medhat Gaber ,"Advances in data stream mining ,*WIREs Data Mining Knowl Discov* 2012.