Analyzing Data Streams Using a Dynamic Compact Stream Pattern Algorithm

Ayodeji Oyewale, Chris Hughes, Mohammed Saraee
School of Computer, Science & Engineering University of Salford, Manchester, United Kingdom
School of Computer, Science & Engineering University of Salford Salford, Manchester, United Kingdom
School of Computing, Science & Engineering Salford University of Salford, Manchester, United Kingdom

***Abstract*—**A growing number of applications that generate massive streams of data need intelligent data processing and online analysis. Data & Knowledge Engineering (DKE) has been known to stimulate the exchange of ideas and interaction between these two related fields of interest. DKE makes it possible to understand, apply and assess knowledge and skills required for the development and application data mining systems. With present technology, companies are able to collect vast amounts of data with relative ease. With no hesitation, many companies now have more data than they can handle. A vital portion of this data entails large unstructured data sets which amount up to 90 percent of an organization's data. With data quantities growing steadily, the explosion of data is putting a strain on infrastructures as diverse companies having to increase their data center capacity with more servers and storages. This study conceptualized handling enormous data as a stream mining problem that applies to continuous data stream and proposes an ensemble of unsupervised learning methods for efficiently detecting anomalies in stream data

Keywords*- Stream data; Steam Mining; Compact data structurres;FP Tree, Path Adjustment Method***.**

I. INTRODUCTION

In recent years, there has been emerging class of data intensive applications such as data processing, traffic monitoring, stock data, and telecom call records which need to process data at a high input-rate. These applications are different from traditional database management system applications where data is at rest with respect to; data arrival rate, frequent rate of update, processing requirements and queries requests.

Data mining which is synonymous to Knowledge Discovery in Data (KDD), has helped diverse business organizations in increasing tangible profit that can be measured in terms of amount of money, number of customers and customers loyalty. It entails analyzing a set of persistent relations, set of well-defined operations, and highly optimized query processing and transaction management component which requires less frequent update and a snapshot of the database is used for processing queries. Data mining consists of five major elements: Extract, transform, and load transaction data onto the data warehouse or management system; Store and manage the data in a multidimensional database system; Provide data access to business analysts and information technology professionals; Analyze the data by application software; Present the data in a useful format, such as a graph or table

II. WHY IS REAL-TIME ANALYSIS CRUCIAL?

Information is emerging as the new currency of business, and remains the basis for true competitive differentiation, innovation, value creation, growth, and risk mitigation. Mere having access to information before industry competition is not enough today. Streaming continuous data allows companies/organizations to analyse data as soon as it becomes available allowing the ability to analysis risks before they occur. Data processing and analytics must occur in real-time and be complemented by real-time action-ability as it can help companies identify new business opportunities and revenue which will eventually result into an increase in profits, acquisition of new customers, and improved service. Above all, it provides a form of security protection as it gives companies a fast way to swiftly associate various event patterns so as to identify relationship.

III. EASE OF USE

Making sense out of stream data has been extremely outstretched as a concern in many fields including telecommunication (Umman Tuğba, & Şimşek Gürsoy,2010; Rashid ,2016; Idris et al.,2012; Bingquan et al., 2012). Therefore, it is stoutly recommended that companies in competition within the same business environment operate their own algorithmic systems fortified with business intelligence and data mining tools to label and isolate an array of incoming data elements for better business advantage. Research in stream data has attracted a great amount of attention in literature due to the gravity of the problem within many organizations and the purpose of making a single pass over data. Consequently, the need to accurately predict an event in near real time before its occurrence is of great importance. The goal of streaming algorithms is to derive profound analysis from a massively long sequence of items taking into account limited time and space constraints

IV. STREAM DATA MINING

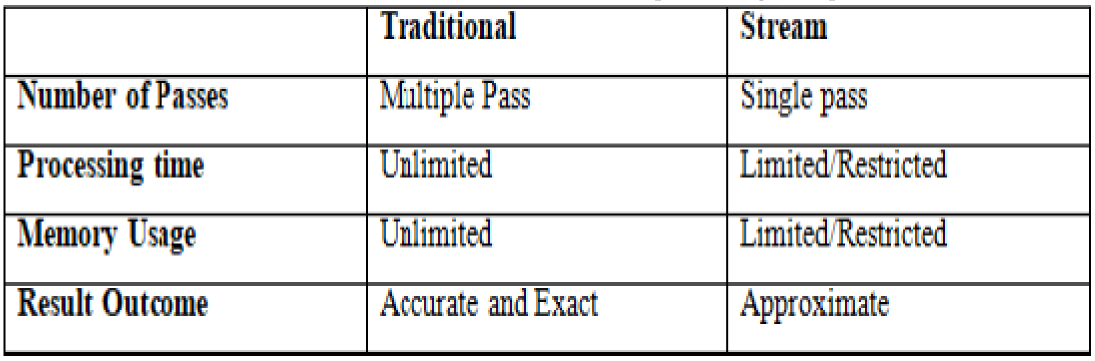
Data Stream mining refers to informational structure extraction as models and patterns from continuous data streams. Information is imperative to make wise and actionable decisions. This is quintessentially true in real life but also in computing and especially critical in several areas, such as finance, fraud detection, business intelligence or battlefield operation. Information flows in from different sources in the form of messages or events, giving a hint on the state at a given time. While data mining usually operates directly on the complete, stored data, stream data tends to make use of supporting application models and data models because of the transient nature of the data to be processed. When dealing with data streams it is usually impossible to store all the data from streams and only a small part of the data can be stored and used for computations within a limited time span. An idyllic procedure for mining data streams should have the following properties: high accuracy, fast adaptation to change and low computation cost in both space and time dimensions i.e. short processing time (Bifet, Holmes, Pfahhringer, Kirkby, Gavalda, 2009). When the volume of the underlying data is very large, it leads to a number of computational and mining challenges.

A. With increasing volume of the data, it is no longer possible to process the data efficiently by using multiple passes. Rather, one can process a data item at most once. This leads to constraints on the implementation of the underlying algorithms. Therefore, stream mining algorithms typically need to be designed so that the algorithms work with one pass of the data

B. In most cases, there is an inherent temporal component to the stream mining process. This is because the data may evolve over time. This behavior of data streams is referred to as temporal locality. Therefore, a straightforward adaptation of one-pass mining algorithms may not be an effective solution to the task. Stream mining algorithms need to be carefully designed with a clear focus on the evolution of the underlying data. Majority of organizations are used to handling data at rest which implies a system that control active transactions and therefore need to have persistence. However, in some cases, those transactions are been executed and analyzed in a data warehouse or data mart. This means that the information is being processed in batch and not in real time.

The main properties of an ideal learning method for mining growing streams of data should have the following: high accuracy and fast adaption to change, low computational cost in both space and time (latency), theoretical performance guarantees, and minimal number of parameters.

TABLE 1: Basic Differences between traditional and stream data processing (Adaptive: João & Pedro, 2004).



	Traditional	Stream
Number of Passes	Multiple Pass	Single pass
Processing time	Unlimited	Limited/Restricted
Memory Usage	Unlimited	Limited/Restricted
Result Outcome	Accurate and Exact	Approximate

The data stream mining task can be considered similar when compared to traditional tasks in terms of targets and objectives reasonably different in terms of data processing; this is so because of the infinite influx of high speed data. This therefore absolutely makes traditional data mining algorithms and methods inept of properly handling data streams.

This can be done in either of two ways;

i. by modifying an existing data/stream mining algorithm to make suitable for stream mining

ii. developing a new stream mining algorithm

C. Analytical Scheme

Using comparably similar approaches employed in data stream mining, there are two general strategies for taking machine learning concepts and applying them to data streams. The *wrapper approach* which aims at maximum reuse of existing schemes, and *adaptation approach* which looks for new methods tailored to the data stream scenery.

Coherent to this research work, the wrapper approach which involves collected of known samples into a batch with the intent to induce a traditional batch learner model. The model(s) required to then be chosen and combined to some degree to form predictions. The impediment of this approach specifically is defining the appropriate training set sizes, and also that training times will be out of the control of a wrapper algorithm, other than the indirect influence of adjusting the training set size. When wrapping around complex batch learners, training sets that are too large could stall the learning process and prevent the stream from being processed at an acceptable speed. Training sets that are too small will induce models that are poor at generalizing to new examples. Memory management of a wrapper scheme can only be conducted on a per-model basis, where memory can be freed by forgetting some of the models that were previously induced.

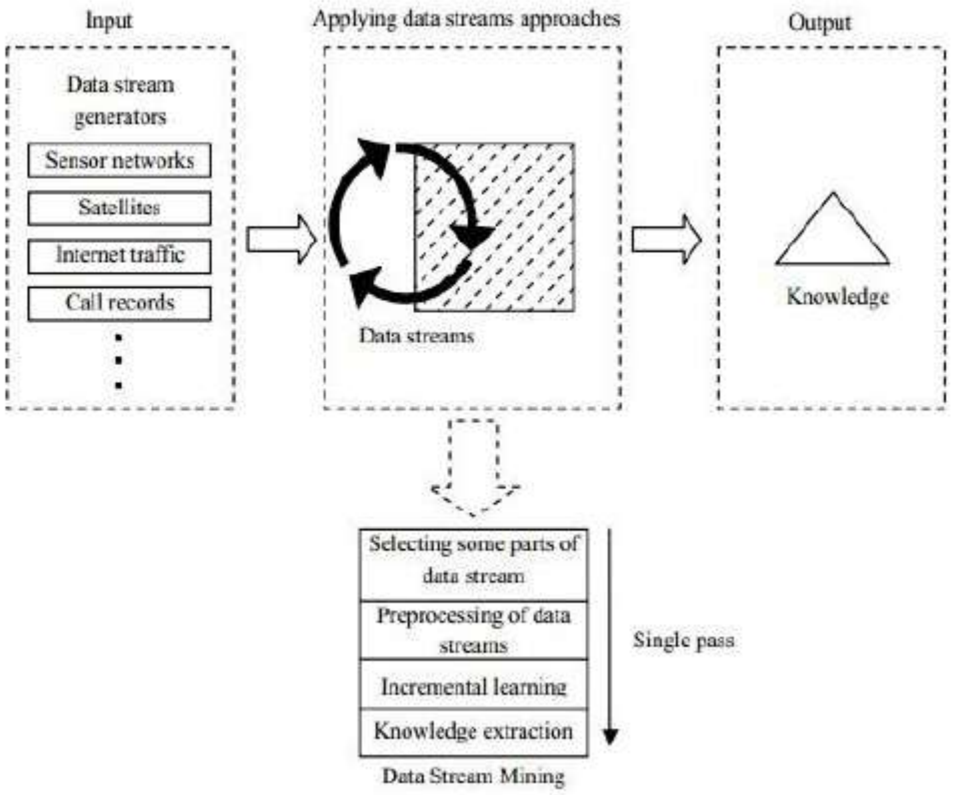


Figure 1. Stream mining process (Adaptive: Mahnoosh Kholghi et al., 2011)

Figure 1 above explains an analytical framework for understanding the steps to interpret a stream mining process. Inputs comprise of various sources that simulates a stream of incoming data, typical examples as seen include sensor networks, satellites feeds, internet traffic etc. An extract, transform, loads (EFL) analysis can be carried out here initially, but this depends on the kind of data being processed. The algorithm presented performs most of its functions at the second stage which include;

i. Selecting part of the dataset using sliding window
ii. Preprocessing of the incoming stream of data with the aid of a middleware
iii. Using iterative method
iv. Ultimately making sense out of the data Thereafter an inference is extracted from the analyzed dataset

V. COMPACT DATA STRUCTURE

In an age where collecting, updating, storing and retrieving of information becomes significant, the introduction of a dynamic, adept and compact data structure is of great essence. This is as a result of the finite memory size and the persistence convergence of the stream data. An efficient and compact data structure is needed to store, update and retrieve the collected information. This is due to bounded memory size and huge amounts of data streams coming continuously. Failure in developing such a data structure will largely decrease the efficiency of the mining algorithm because, even if we store the information in disks, the input and output operations will increase the processing time. The data structure needs to be incrementally maintained since it is not possible to rescan the entire input due to the huge amount of data. In [Chen *et al.*, 2007], the authors employ a prefix tree data structure to store item ids and their support values, block ids, head and node links pointing to the root or a certain node. In [Giannella, 2003], a FP-tree is constructed to store items, support information and node links. A thorough and unmitigated data structure is a determinant in developing an efficient algorithm. Since it is directly associated with the way we handle newly arrived information and update old stored information. A small and compact data structure which is efficient in inserting, retrieving and updating information is most favorable when developing an algorithm to mine association rules for stream data.

VI. PROCEDURES IN ANALYZING STREAM DATA

In order to develop and analyse an incessant influx of data, it is required to take into account vital information with respect to stream data.

1) Data preprocessing:

Information preprocessing in this examination work includes the utilization of a middleware which fills in as a working shrouded interpretation layer which empowers a correspondence, collaboration, and information administration between the working framework and the application running on it.

Data pre-processing consumes most of the time in the knowledge discovery process. The challenge here is to automate such a process and integrate it with the mining techniques

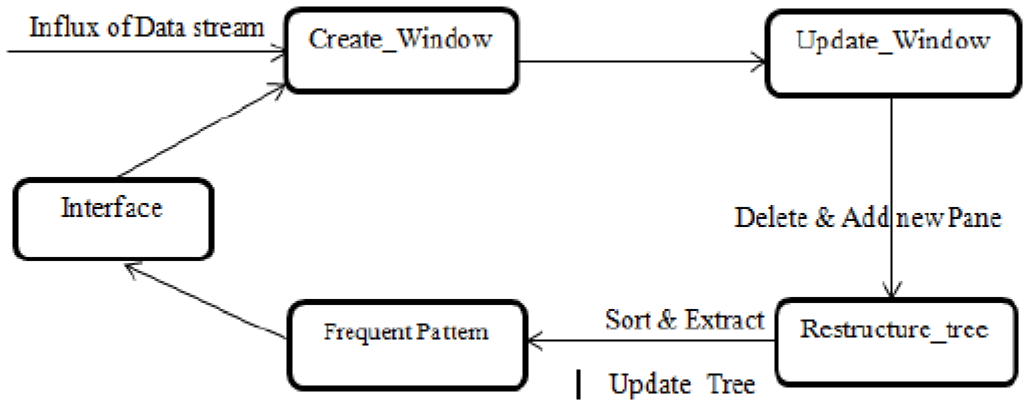


Figure 2: Data-preprocessing

2) The middleware:

The middleware class is made to assemble skeleton of communication between the system and the information it is intended to be processed. The constructor of this class requires record name of the document that ought to be prepared, the thing column(s) that the calculation should process and the value-based section the thing ought to be coordinated against.

```
def __init__(self, file, transaction_field, item_field):
```

The middleware class has a function call *format_data* which is used to handle data that returns the algorithms specific data type. The middleware also declares an abstract method named *process_data* that must be implemented by every subclass of the class

3) Sliding Window:

A sliding window algorithm places a buffer between the application program and the network data flow. For most applications, the buffer is typically in the operating system kernel, but this is more of an implementation detail than a hard-and-fast requirement. The sliding window technique inspects every time window at all scales and location over a time stamp which means our data will be classified according to the most recent item set provided. Typically, sliding window algorithms serve as a form of flow control for data transfers. Datasets in a sliding window are often described in a structure

$$(Di) = \{So, S1 \ldots Si\} i \leq n - 1$$
$$\{Si, Si+1 \ldots Sn + i - 1\} \geq n - 1 \quad (1)$$

If the timespan (length, l) of a window is denoted with *n*

However; data at a point Ti in the window is denoted by

Where; Di: Dataset present in the sliding window

Si: Data values of the dataset at the point i

VII. DATA MODELS

When weighed against data in traditional databases, data streams are unbounded and the number of transactions increases over time. As a result of these, different data models for effective processing have been suggested in different mining algorithms. The adapted model is based on a sliding window. That is, despite the infinite arrival of the stream data, the frequent itemsets are derived based on the most recent data that is being captured within a stipulated sliding window where the present time signifies end point of that window.

One justification for such a sliding-window model is that due to temporal locality, the data in streams is bound to change with time, and many a times people are interested in the most recent array and patterns from the stream data(2).

Data streams differ from the conventional stored relation model in several ways:

i.) The data elements in the stream arrive online.
ii) The system has no control over the order in which data elements arrive to be processed, either within a data stream or across data streams.
iii) Data streams are potentially unbounded in size.
iv) the instance an element from a data stream has been processed it is discarded or archived — unless specifically stored in an external storage point it cannot be retrieved easily, which ideally is small relative to the size of the data streams. Frequently, information stream inquiries may perform joins between information streams and put away social information.

For the motivations behind this paper, we will accept that if put away relations are utilized, their substance stay static. Therefore, we block any potential exchange preparing issues that may emerge from the nearness of updates to put away relations that happen simultaneously with information stream handling.

VIII. *CLAIMS IN ANALYZING STREAM DATA*

Essentially, the process of data stream requires two layers
i. A storage layer
ii. A processing layer

The prerequisite of the storage layer is to provide and maintain a detailed prioritization and strong consistency to enable fast, inexpensive, and repayable reads and writes of substantial streams of data. The processing layer is responsible for continually using up data from the storage layer, running computations on that data, and then notifying the storage layer to delete data that is no longer needed or deleting the limited storage area for a new set of data to come in.

IX. STREAM DATA ALGORITHM

The algorithms for processing streams one way or the other involves a form of summarization of the stream by making useful sample of the stream to eliminate most of the "undesirable" elements. The speed of the mining algorithm must be faster than the incoming data rate otherwise, summarization is of great effect. An algorithm can also run faster by processing less information, either by stopping early or storing less, thus having less data to process. The more time an algorithm has, the more likely it is that accuracy can be increased. The performance of an algorithm that operates on data streams is measured by three basic factors:

a. The number of passes the algorithm must make over the stream.
b. The available memory.
c. The running time of the algorithm

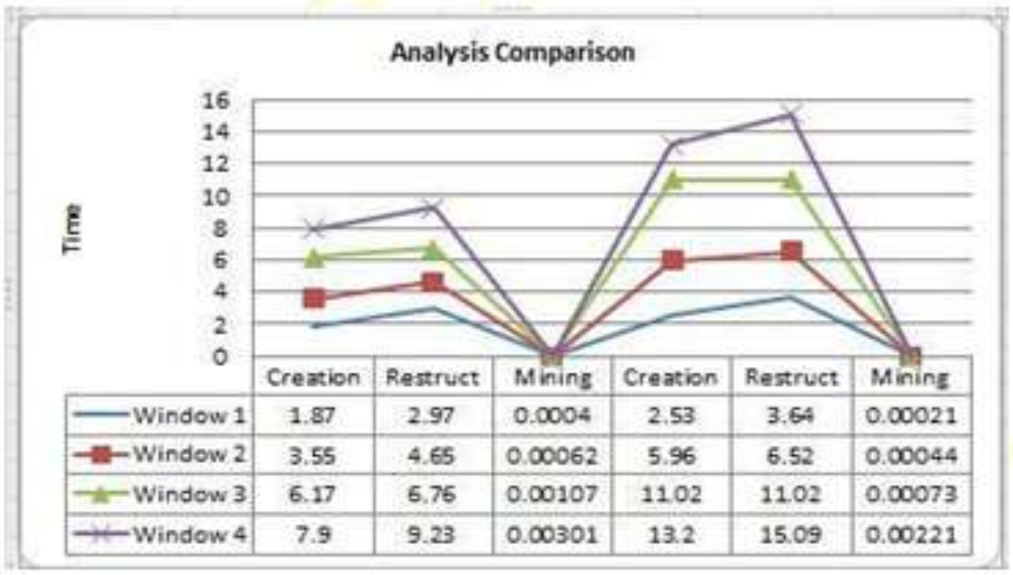
X. RESULT ANALYSIS

A pragmatic evaluation of the completion of CSP implementation is made. A comparative analysis of CSP tree with FP Stream algorithm is done using pyrhon programming language. All programs are done using Python 3.0 and executed in Winsows 7 on a 2.66 GHz CPU with 1GB memory usage. And this is done using the BSM, Sales and Telecom datasets and for further enumeration a call record datasets from Kaggle was also used.

TABLE II: DATASET CHARACTERISTICS

Datasets	No of Transactions	Size(M)
BSM(Click stream)	515,597	2GB
Sales	88, 475	9.7M
Telecoms	9,683,900	10.6GB
Call record	37,283,928	43GB

The aforementioned Table I above shows the characteristics of the datasets used for the analysis. The BSM dataset contains several entries from an electronics retailer. The Sales dataset consists of retail Watson Analytics Sample Data of Sales Products and the telecoms datasets consists of the call detail record of customers. In this experiment, the average runtime of all active windows are computed for the two algorithms on each data. At every window, mining is perfromed on the recent window. The pane size is dependent on the size of the dataset introduced.



	Creation	Restruct	Mining	Creation	Restruct	Mining
Window 1	1.87	2.97	0.0004	2.53	3.64	0.00021
Window 2	3.55	4.65	0.00062	5.96	6.52	0.00044
Window 3	6.17	6.76	0.00107	11.02	11.02	0.00073
Window 4	7.9	9.23	0.00301	13.2	15.09	0.00221

Figure 3: CSP AND FP Stream Comparison on BSM Data

A comparative analysis for BMS click stream dataset between CSP Tree and FP Stream tree is shown in the Figure 3 above. The graph indicates precisely that at each level of data creation, restructuring and mining, CSP tends to outpace FP Stream Tree.

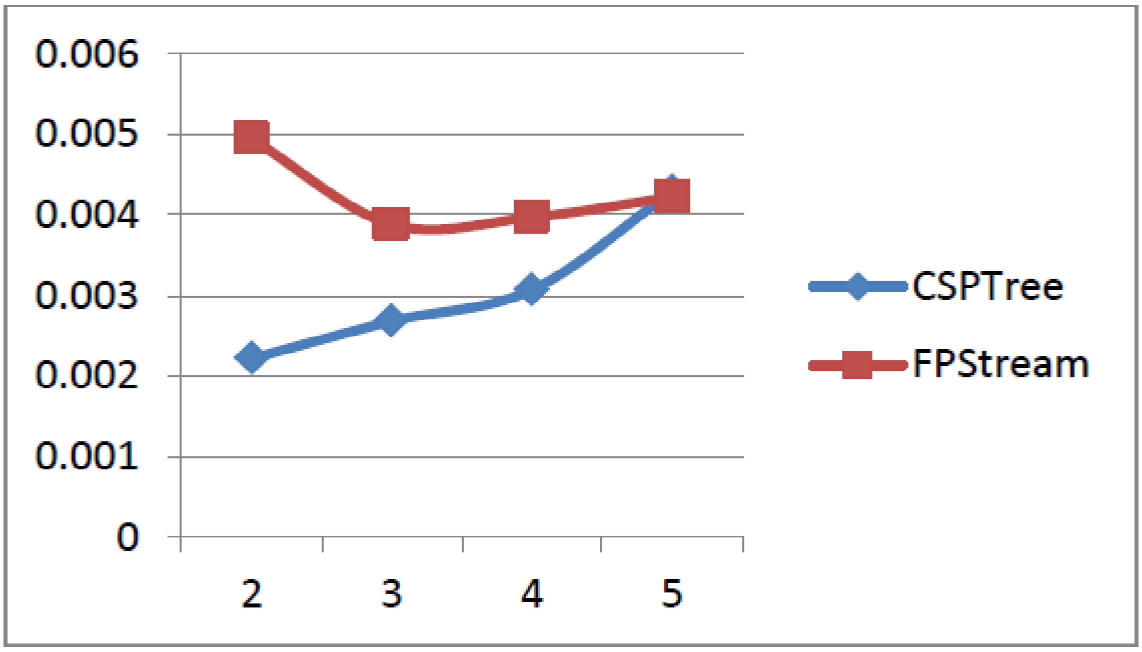


Figure 4: CSP AND FP Stream Comparison on Sales Data

Also shown above is the time it takes to restructure each incoming tree for the two algorithms. In the first window, FP tree restructures faster but in subsequent windows CSP Tree outperforms it. The following datasets are being analysed with respect to the time of tree creation, the time rate for restructuring and how long it takes to mine the data in conjunction with the varying time window. This juxtapose has been made to make a comparative analysis between CSP and FP algorithm. The time it takes to create a tree using CSP algorithm is shorter when compared to FP algorithm. It takes CSP 1.8sec to build a tree while it takes FP algorithm over 3 sec. The figure 5 below shows a graphical advantage of CSP over FP

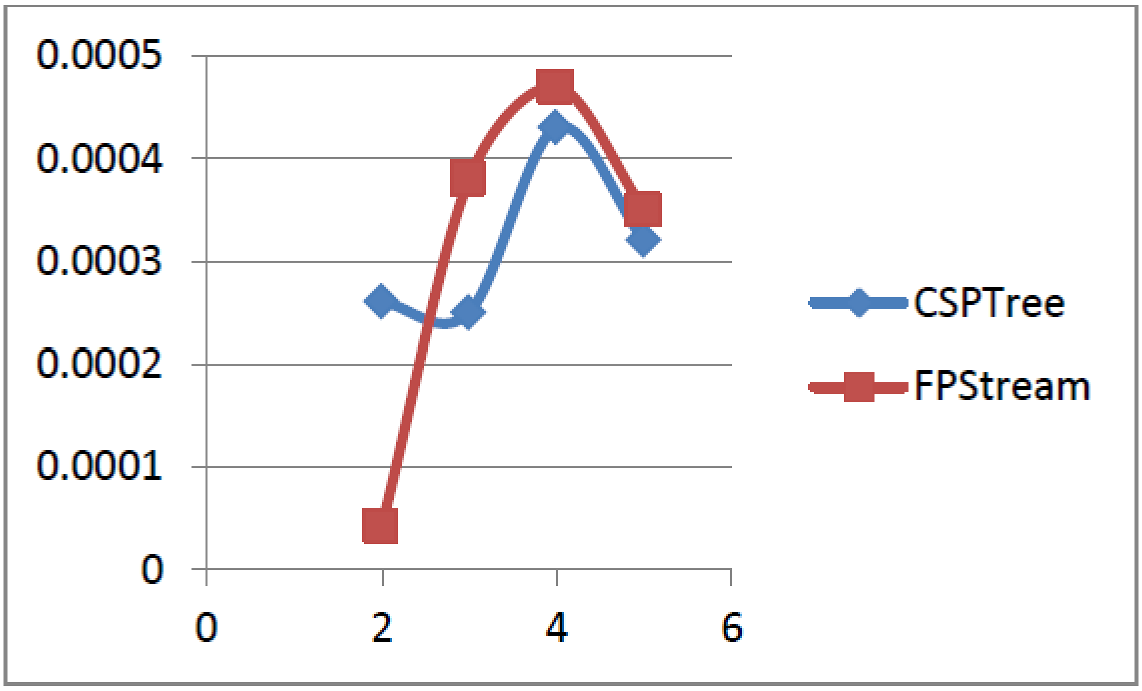


Figure5. Mining time analysis between CSP and FPstream over Call Records dataset

The time it takes for each algorithm to mine the call record dataset is shown in the figure 4 above. At the 2nd window, it takes FPStream twice the time of CSP to analyse the incoming datasets and as each window progresses, CSP tends to be more time efficient compared to FPstream.

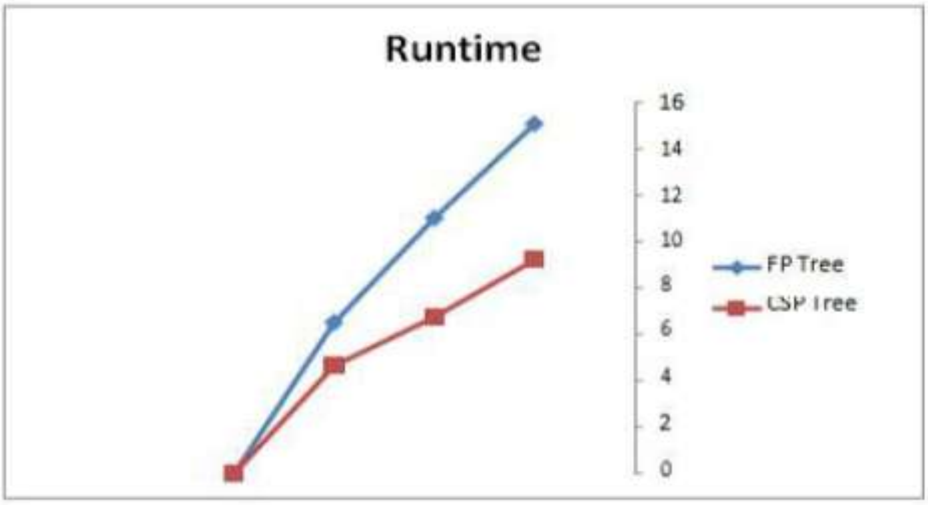


Figure 6: Graphical Advantage of CSP over FP Stream

The above Figure 5 depicts the Tree restructuring phase of the all dataset. While restructuring the tree, we use the Path Adjustment method (PAM) which depends on the Degree of Displacement of two items and it swaps two nodes with Bubble sort method. And also, the branch sort algorithm makes use of the merge sort approach. Due to the high rate of displacement while using PAM, it is unsuitable for the algorithm hence BSM performs better during tree restructuring approach with merge algorithm. The merge sort method unlike quicksort makes use of Divide and Conquer algorithm. The most recent input array is intermittently divided in two halves; it sorts the two halves and then merges the two sorted halves.

XI. CONCLUSION

We have proposed CSP-tree that dynamically achieves frequency-descending prefix tree structure with a single-pass by applying tree restructuring technique and considerably reduces the mining time. We also adopted Branch sorting method using merge sort which is a new tree restructuring technique and presented guideline in choosing the values for tree restructuring parameters. It shows that despite additional insignificant tree restructuring cost, CSP-tree achieves a remarkable performance gain on overall runtime. The algorithm fulfils the requirements necessary for coping with data streams while remaining efficient, an achievement that was rare prior to its introduction. The easy-to-maintain feature and property of constantly summarizing full data stream information in a highly compact fashion facilitate its efficient applicability in interactive, incremental and stream data.

REFERENCES

[1]. E. Ascarza, P. Ebbes, O. Netzer and M. Danielson, Beyond the Target Customer: Social Effects of CRM Campaigns.2016.

[2]. Babcock, S. Babu, M. Datar, R. Motwani, J. Widom, Models and issues in data stream systems. In:Proceedings of 21st Acmigmod-Sigact-Sigart symposium on principles of database systems(PODS' 02), pp 1–16. 2002.

[3]. B. Borja, C. Bernardino, C. Alex, R. Gavald`a, M. David Manzano-Macho, The Architecture of a Churn Prediction System Based on Stream Mining. 2013.

[4]. Intel IT, IT Best Practices Business IntelligenceRetrieved from http://www.intel.com/it/.Feb 2012.

[5]. J.-L. Koh and S.-F. Shieh. An efficient approach for maintaining association rules based on adjusting FP-tree structures. In Lee Y-J, Li J, Whang K-Y, Lee D (eds) Proc. of DASFAA 2004. Springer-Verlag, Berlin Heidelberg New York, 417–424. , 2004

[6]. S. K. Tanbeer, C. F. Ahmed, B. S. Jeong,, and Y.-K. Lee. CP-tree: A tree structure for single-pass frequent pattern mining. In Proc. of PAKDD, Lecture Notes Artif Int, 1022-1027. 2008

[7]. T.A. Rashid, Convolutional Neural Networks based Method for Improving Facial Expression Recognition. In: Corchado Rodriguez J., Mitra S., Thampi S., El-Alfy ES. (eds) Intelligent Systems Technologies and Applications 2016. ISTA 2016. Advances in Intelligent Systems and Computing, vol 530. Springer, Cham. Oct 2016.