International Journal of Electronics, Communication & Soft Computing Science and Engineering
ISSN: 2277-9477, Volume 2, Issue 9

An Approach towards Mining Frequent Items and Item Sets from Distributed Data Stream

Ms. Poonam. A. Manjare **Mrs. R. R. Shelke**

***Abstract*- Data mining is an increasingly important technology for extracting useful knowledge hidden in large collections of data. The proposed work presents the design and the implementation of architecture for the analysis of data streams in distributed environments. In particular, data stream analysis has been carried out for the computation of items and item sets that exceed a frequency threshold. The mining approach is hybrid, that is, frequent items are calculated with a single pass, using a sketch algorithm, while frequent item sets are calculated by a further multi-pass analysis. The architecture combines parallel and distributed processing to keep the pace with the rate of distributed data streams. In order to keep computation close to data, miners are distributed among the domains where data streams are generated**.

***Keywords*- Data Mining, Frequent Item, Frequent Item sets, Data Streams.**

I. INTRODUCTION

Mining data streams is a very important research topic, because in many cases data is generated by external sources so rapidly that it may become impossible to store it and analyze it offline. Moreover, in some cases streams of data must be analyzed in real time to provide information about trends, outlier values or regularities that must be signaled as soon as possible. The need for online computation is a notable challenge with respect to classical data mining algorithms [1-2]. Important application fields for stream mining are as diverse as financial applications, network monitoring, security problems, telecommunication networks, Web applications, sensor networks, analysis of atmospheric data, etc. A further difficulty occurs when streams are distributed, and mining models must be derived not only for the data of a single stream, but for the integration of multiple and heterogeneous data streams [3]. This scenario can occur in all the application domains mentioned before. For example, in a Content Distribution Network, user requests delivered to a Web system can be forwarded to any of several servers located in different and possibly distant places, in order to serve requests more efficiently and balance the load. In such a context, the analysis of user requests, for example to discover frequent patterns, must be performed with the inspection of data streams detected by different servers. Frequent item set mining is a core data mining operation and has been extensively studied over the last decade [4-8]. Algorithms for frequent item set mining form the basis for algorithms for a number of other mining problems, including association mining, correlations mining, and mining sequential and emerging patterns [6].

Two important and recurrent problems regarding the analysis of data streams are the computation of frequent items and frequent item sets from transactional datasets. The first problem is very popular both for its simplicity and because it is often used as a subroutine for more complex problems. The goal is to find, in a sequence of items, those whose frequency exceeds a specified threshold. When the items are generated in the form of transactions, sets of distinct items, it is also useful to discover frequent sets of items. A k-item set, i.e., a set of k distinct items, is said to be frequent if those items concurrently appear in a specified fraction of transactions.

II. LITERATURE REVIEW

The analysis of data streams has recently attracted a lot of attention owing to the wide range of applications for which it can be extremely useful. Important challenges arise from the necessity of performing most computation with a single pass on stream data, because of limitations in time and memory space. Stream mining algorithms deal with problems as diverse as clustering and classification of data streams, change detection, stream cube analysis, indexing, forecasting, etc [14]. In the propose work, a major need is to identify frequent patterns in data streams, either single frequent elements or frequent sets of items in transactional databases. A rich survey of algorithms for discovering frequent items is provided by Cormode and Hadjieleftheriou [9]. In proposed work, the discussion focuses on the two main classes of algorithms for finding frequent items. Counter-based algorithms have their foundation on some techniques proposed in the early 80s to solve the Majority problem [15], i.e., the problem of finding a majority element in a stream, using a single counter. Variants of this algorithm were devised, sometimes

decades later, to discover items whose frequencies exceed any given threshold. LossyCounting is perhaps the most popular algorithm of this type [11]. The second class of algorithms computes a sketch, i.e., a linear projection of the input, and provides an approximated estimation of item frequencies using limited computing and memory resources. Popular algorithms of this kind are CountSketch [16] and CountMin [13], and the latter is adopted in this proposed work. Advantages and limitations of sketch algorithms are discussed in [19]. Important advantages are the notable space efficiency (required space is logarithmic in the number of distinct items), the possibility of naturally dealing with negative updates and item deletions, and the linear property, which allows sketches of multiple streams to be computed by overlapping the sketches of single streams. The main limitation is the underlying assumption that the domain size of the data stream is large, but this is true in many significant domains. Even if modern single-pass algorithms are extremely sophisticated and powerful, multi-pass algorithms are still necessary either when the stream rate is too rapid, or when the problem is inherently related to the execution of multiple passes, which is the case, for example, of the frequent item sets problem. Single-pass algorithms can be forced to check the frequency of 2- or 3-itemsets, but this approach cannot be generalized easily, as the number of candidate *k*-item sets is combinatorial, and it can become very large when increasing the value of *k* [9]. Therefore, a very promising avenue could be to devise hybrid approaches, which try to combine the best of single- and multiple-pass algorithms. A strategy of this kind, discussed in [12], is adopted in the mining architecture presented in this proposed work. The analysis of streams is even more challenging when data is produced by different sources spread in a distributed environment. A thorough discussion of the approaches currently used to mine multiple data streams can be found in [17]. The proposed work distinguishes between the centralized model, under which streams are directed to a central location before they are mined, and the distributed model, in which distributed computing nodes perform part of the computation close to the data, and send to a central site only the models, not the data. Of course, the distributed approach has notable advantages in terms of degree of parallelism and scalability. An interesting approach for the continuous tracking of complex queries over collections of distributed streams is presented in [3]. To reduce the communication overhead, the adopted strategy combines two technical solutions: (i) remote sites only communicate to the coordinator concise summary information on local streams (in the form of sketches); (ii) even such communications are avoided when the behavior of local streams remains reasonably stable, or predictable: updates of sketches are only transmitted when a certain amount of change is observed locally. The success of this strategy depends on the level of approximation on the results that is tolerated. A similar approach is adopted in [18]: here stream data is sent to the central processor after being filtered at remote data sources. The filters adapt to changing conditions to minimize stream rates while guaranteeing that the central processor still receives the updates necessary to provide answers of adequate precision.

III. MINING FROM DISTRIBUTED DATASET

Mining data streams is a very important, because in many cases data is generated by external sources so rapidly that it may become impossible to store it and analyze it offline. Moreover, in some cases streams of data must be analyzed to provide real time information. These challenges have led to the formalization of the continuous, distributed, streaming model.

Following are the steps of mining items

A. Data Fetching

In this data would fetch from database. Mining data is very important because data generated by external sources must be analyzed in real time for the computation of item and item sets.

B. Data Analysis

This would used to analyze the fetch data and find the data which the user had searched previously. The stream mining architecture aims at solving the problem of computing frequent items

and frequent item sets from distributed data streams. It is assumed that stream sources, though belonging to different domains, are homogenous, so that it is useful to extract knowledge from their union. Typical cases are the analysis of the traffic experienced by several routers of a wide area network, or the analysis of client requests forwarded to multiple web servers.

C. Fuzzy Set Distribution

Here data would be distributed in fuzzy sets. A fuzzy set provides a natural basis for the theory of possibility.

D. Recommendation

Here data would be recommending through fuzzy sets.

E. Result Evaluation and system operation

Result evaluation would be done through fetched data from database, analysis of fetched data which determine frequent item and item sets and distributed data from fuzzy sets.

IV. CONCLUSION

The distributed stream mining system is a contribution in the field and it aims at solving the problem of computing frequent items and frequent item sets from distributed data streams by exploiting a hybrid single pass/multiple-pass strategy. We assumed that stream sources, though belonging to different domains, are homogenous, so that it is useful to extract knowledge from their union. Beyond presenting the system architecture, we described a prototype that implements it and discussed a set of experiments performed in a real Grid environment. The experimental results confirm that the approach is scalable and can manage large data production by using an appropriate number of miners in the distributed architecture.

REFERENCES

[1] M. M. Gaber, A. Zaslavsky, and S. Krishnaswamy, "Mining data streams: A review," *ACM SIGMOD Record*, vol. Vol. 34, no. 1, 2005.

[2] C. C. Aggarwal, *Data Streams: models and algorithms*. Springer, 2007.

[3] G. Cormode and M. Garofalakis, "Approximate continuous querying over distributed streams," *ACM Transactions on Database Systems*, vol. Vol. 33, no. 2, 2008.

[4]R. Agrawal, H. Mannila, R. Srikant, H. Toivonent, and A. Inkeri Verkamo. Fast discovery of association rules. In U. Fayyad and et al, editors, *Advances in Knowledge Discovery and Data Mining*, pages 307–328. AAAI Press, Menlo Park, CA, 1996.

[5]B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom. Models and Issues in Data Stream Systems. In *Proceedings of the 2002 ACM Symposium on Principles of Database Systems (PODS 2002) (Invited Paper)*. ACM Press, June 2002.

[6]J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In *Proceedings of the ACM SIGMOD Conference on Management of Data*, 2000.

[7] Mohammed J. Zaki. Parallel and distributed association mining: A survey. *IEEE Concurrency*, 7(4):14 – 25, 1999.

[8]Z. Zheng, R. Kohavi, and L. Mason. Real World Performance of Association Rule Algorithms. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 401–406. ACM Press, August 2001.

[9]R. Jin and G. Agrawal, "An algorithm for in-core frequent itemset mining on streaming data," in *5th IEEE International Conference on Data Mining ICDM*, Houston, Texas, USA, 2005, pp. 210–217.*Conference on Data Mining ICDM*, Houston, Texas, USA, 2005, pp. 210–217.

[10]G. Cormode and M. Hadjieleftheriou, "Finding the frequent items in streams of dat," *Communications of the ACM*, vol. 52, no. 10, pp. 97–105, 2009.

[11]A. Wright, "Data streaming 2.0," *Communications of ACM (CACM)*, vol. Vol. 53, no. 4, 2010.

[12]G. Manku and R. Motwani, "Approximate frequency counts over data streams," in *International Conference on Very Large Data Bases*, 2002.

[13] G. Cormode and S. Muthukrishnan, "An improved data stream summary: The count-min sketch and its applications," *J. Algorithms*, vol. Vol. 55, 2005.

[14]C. Aggarwal, "An introduction to data streams," in *Data Streams: Models and Algorithms*, C. Aggarwal, Ed. Springer, 2007, pp. 1–8.

[15] M. Fischer and S. Salzburg, "Finding a majority among n votes: solution to problem 81-5," *J. Algorithms*, vol. 3, no. 4, pp. 376–379, 1982.

[16]M. Charikar, K. Chen, and M. Farach-Colton, "Finding frequent items in data streams," in *Proceedings of the International Colloquium on Automata, Languages and Programming (ICALP)*, 2002.

[17] A. G. Srinivasan Parthasarathy and M. E. Otey, "A survey of distributed mining of data streams," in *Data Streams: Models and Algorithms*, C. Aggarwal, Ed. Springer, 2007, pp. 289– 307.

[18] C. Olston, J. Jiang, and J. Widom, "Adaptive filters for continuous queries over distributed data streams," in *SIGMOD '03: Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, San Diego, California, 2003.

[19] C. C. Aggarwal and P. S. Yu, "A survey of synopsis construction in data streams," in *Data Streams: Models and Algorithms*, C. Aggarwal, Ed. Springer, 2007, pp. 169–207.

AUTHOR'S PROFILE

Ms. Poonam. A. Manjare
M.E Second Year,
Department of computer science
And Engineering

H.V.P.M C.O.E.T., Amravati, India.
manjarepoonam@gmail.com

Mrs. R. R. Shelke
Department of computer science
And Engineering
H.V.P.M C.O.E.T., Amravati, India.
rajeshrishelke@rediffmail.com