Improvement of Apriori Algorithm for Missing Itemset Identification and Faster Execution

Anjan Dutta*
Techno International NewTown, Kolkata, INDIA

Runa Ganguli*
The Bhawanipur Education Society College, Kolkata, INDIA

Punyasha Chatterjee†
Jadavpur University, Kolkata, INDIA

Narayan C. Debnath‡
Eastern InternationalUniversity,
Thu DauMot, VIETNAM

Soumya Sen§
University of Calcutta, Calcutta, INDIA

Abstract

Association rule mining (ARM) is an important data mining strategy to analyze the relationship among the items. Apriori algorithm is the most used approach to implement association rule mining. We identified two major issues of Apriori. Apriori follows an iterative approach consisting of multiple database scans for searching frequent itemsets that satisfy certain threshold criteria. The same predefined threshold value is maintained throughout the repetitive stages of the Apriori method and hence there is a huge possibility of discarding higher-order itemsets, though all of its sub-itemsets are frequent. Some of these ignored itemsets if used intelligently have a huge potential for business value addition. Furthermore, in the Apriori procedure, an exponential number of computations is required to check whether an item is important or not and that makes the entire pattern mining system costly. In this study first, we identify the hidden business-critical item sets that are otherwise ignored in the traditional Apriori process. Furthermore, a novel approach is proposed here to evaluate whether an item is interesting or not at a considerably reduced computational time.

*Email: anjan.dutta.edu@gmail.com, runa.ganguli@gmail.com,
†School of Mobile Computing and Communication, Email: punyasha.chatterjee@gmail.com.
‡School of Computing and Information Technology. Email: narayan.debnath@eiu.edu.vn.
§A.K. Choudhury School of Information Technology. Email: iamsoumyasen@gmail.com.

Key Words: Association rule mining, apriori algorithm, data mining, missing itemset, execution time.

1 Introduction

Data mining is an integral part of any business in modern day to improve the performance in terms of profit, sales, forecasting etc. It is comprised of extracting the data, analyzing the data and then generating a report or pattern to ease out the business process. Data mining, also known as Knowledge Discovery in Databases (KDD) [5], is the process of discover-ing hidden and interesting patterns from a huge amount of data for making essential business-oriented decisions. Association rule mining [2, 6, 8] is a popular technique in data mining to analyze the relationship among the different items in a set of transactions. It is conceptualized as, for every occurrence of A there exists certain numbers of occurrence of B in any trans-action database. Knowledge of the frequent sets is generally used to design association rules stating how a set of items (itemset) influences the presence of another itemset in the transaction database. The mining rules are more applicable and useful in the market basket analysis. Association rules are frequently used in business intelligence [12] to help their marketing, advertisement, inventory control, fault prediction, product recommendation etc. Due to its huge business scope, association rule mining is a well-studied research problem among the researchers. Among the several association rule mining techniques Apriori algorithm [2] is the most studied and widely used algorithm for Frequent Pattern Mining (FPM) [6, 9, 13].

Apriori algorithm is used to mine all frequent itemsets in a transaction database. This algorithm begins with defining the support of an item that is the frequency of the occurrence of the items or itemsets in the transactional dataset. An itemset of size k whose support is greater than some user-specified minimum support threshold is said to be a frequent itemset and denoted by L_k, otherwise the items are infrequent. Any candidate itemset of size k, denoted by C_k is a potentially frequent itemset. The algorithm begins by scanning the whole database to find the set of frequent 1-itemsets by counting each item in database. The resulting set is called L_1 which is used to determine the set of frequent 2-itemsets which in turn is used to find the set of frequent 3-itemsets and so on until no more frequent k-itemsets can be found. In this way, it uses an iterative level-wise searching approach where k-itemsets are used to generate (k+1) itemsets. If there are n items, we can generate 2^n numbers of possible combination as given in Figure 1. To reduce the search space and improve the efficiency of this level-wise frequent itemset generation, the concept of pruning is used. This Apriori property states that if an itemset is not frequent, any large subset from it is also non-frequent [2]. This condition leads to pruning of some candidate itemsets from the search space in the database. It is shown in Figure 2.

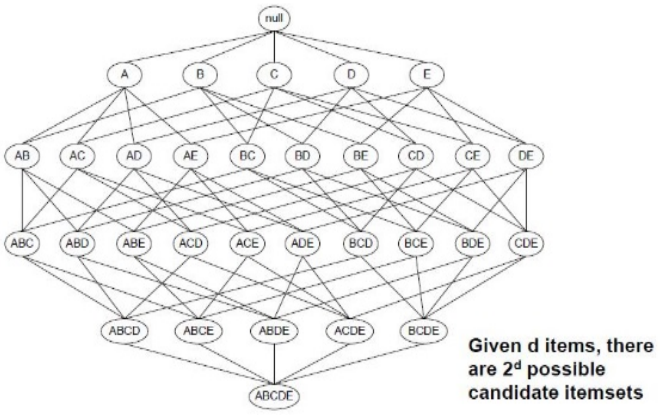

Figure 1: Frequent itemset generation

Although Apriori is a widely used technique for association rule mining several limitations could also be identified. In Apriori, itemsets are discarded based on the given threshold value and that value is also fixed for all the higher order itemsets. This static assumption leads to the discard of some of the interesting patterns by pruning. Another problem of Apriori is high time complexity as every new level or high order itemset (k+1) is generated from the k itemset.

In this paper, we address the above two limitations and propose new methodologies to find out the interesting missing patterns that are pruned by Apriori and also speedup the computation for quick decision making.

The rest of the paper is organized as follows: Section 2 presents the related studies on the different improvised version of Apriori Algorithm and its applications. Section 3 discusses the motivation of the work. In Section 4, the methodology of our work is described and the results are discussed in Section 5. The paper is concluded in Section 6.

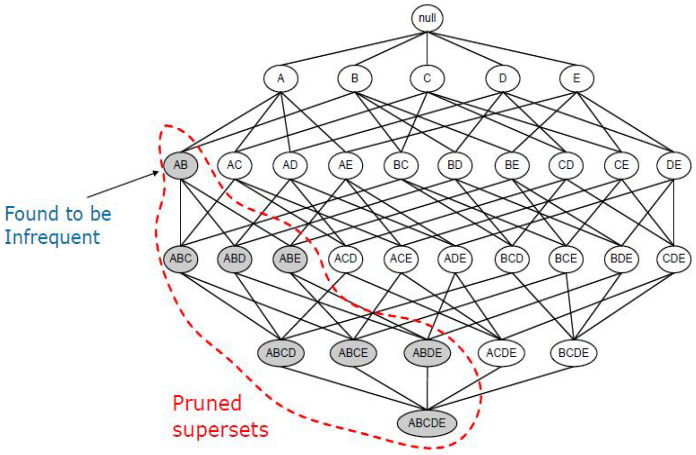

Figure 2: Apriori Principle of reducing number of candidates

2 Related Work

Apriori algorithm is one of the most widely used technique in data mining for frequent itemset calculation. However, it has many drawbacks and these are critical to large dataset mining. As a result, many researchers over the years have pointed out these problems and proposed different versions of improved Apriori algorithm. In [15], a Frequent Pattern (FP) Growth ARM algorithm has been presented that removed the disadvantages of traditional Apriori and proved to be efficient in terms of number of database scan and time. It [15] compresses a large database into a compact data structure Frequent Pattern (FP) tree [10] which is based on FP-Growth algorithm [10] and by recursively searching the tree, all frequent patterns are found. The authors have shown that FP-Growth algorithm outperforms the classical Apriori algorithm in terms of running time, number of database scan, but memory consumption is relatively high. Reducing thenumber of dataset scans for enhancing efficiency of the algorithm have been shown by many researchers [11, 16, 18]. Another approach is seen in [3], where instead of scanning the whole database for frequent itemsets, the authors have focussed on selective scanning of only some specific transactions based on minimum support count. It is seen by experiment that the total number of scanned transaction for candidate itemset generation is less when the same database uses standard Apriori and the improved Apriori [3] reduces the time consumed by 67.38% in comparison with the original Apriori. Singh et. al [17] have also worked towards reducing the database scan time by cutting down unnecessary transaction records and redundant sub-items generation during the pruning stage. The method eliminates the generation of candidates those having infrequent subset i.e., the itemsets having support count less than the specified threshold. The authors have proposed an optimized method for Apriori algorithm which reduces the size of the database by introducing an extra attribute Size_Of_Transaction

(SOT), containing the number of items in individual transaction in database. The improved algorithm [17] not only optimizes the algorithm for reducing the size of the candidate set of k-itemsets, Ck, but also reduces the I/O spending by cutting down transaction records in the database. However, it has the overhead to manage the new database after every generation of frequent set of k-itemsetsLk.

Minimizing the candidate generation is important for increasing efficiency of Apriori algorithm. Factors such as set size and set size frequency are being used [1] to eliminate non-significant candidate itemsets. The author implemented both the original and modified algorithms and average results favoured the modified algorithm by 38% and 33% in terms of execution time and database pass respectively. Apriori-Improve algorithm [4] was mainly proposed to optimize 2-itemset generation and transactions compression. The authors have used hash structure for frequent 2-itemset (L_2) generation directly from one database scan without generation of C_1, L_1 and C_2. The searching cost is also reduced by replacing hash tree by hash table. The algorithm used an efficient horizontal data representation and optimized strategy for saving time and storage space. In another research work [19] two major bottlenecks of FIM were addressed. These are: multitude of candidate 2-itemsets(C_2) and the poor efficiency of counting their support. The proposed algorithm, Reduced Apriori Algorithm with Tag (RAAT), reduces one redundant pruning operation of C_2. It is shown here how the use of transaction tag helps to speed up support calculation. The paper concludes that for relatively small support, RAAT algorithm runs faster than the traditional one. One more enhanced version of Apriori namely DCP (Direct Count of candidates & Prune transactions) [14] was proposed that focused on optimizing the initial iterations of Apriori when datasets are characterized by mainly short or medium length frequent patterns. The main enhancements include database pruning techniques, use of an effective method for storing candidate itemsets and their support counting. Application of improved Apriori algorithm was presented in [7] over a mobile e-commerce recommendation system. The approach converts the transaction database into a corresponding binary matrix to accelerate the algorithm efficiency, initially filtering out unrelated data in the candidate sets and hence improving the mining efficiency too.

3 Motivation and Objective

In the traditional Apriori method a significant number of insignificant items or itemsets having ignorable support count, are generated in the interim stages. Therefore, a pruning strategy is taken to eliminate those item sets. The item sets are eliminated based on two conditions-

If one of the sub-itemsets is infrequent then the itemset is infrequent.

If the support count of the itemset is less than the threshold then the itemset should be pruned.

Example 1: Let us consider a 3-itemset {p, q, r} database whose support count is less than the threshold support count s whereas the support count of all its 2-itemset subset {p, q}, {q, r} and {p, r} is greater than or equal of s. As per the approach of traditional Apriori although all the subset of a set is considered as frequent itemset, the superset {p, q, r} may become infrequent.

In the above approach, there may be itemsets having frequent sub-itemsets though its support count is less than the threshold value. If the sub-itemsets are valuable then keeping these itemsets together may add value to the business e.g. number of purchases of this larger set might increase resulting in the overall business growth.

It is because Apriori algorithm considers static threshold value for all k-itemset. In this example the itemset {p, q, r} can be an interesting pattern that has the potential to add value to the business. In this study, these kinds of valuable patterns that are otherwise ignored in the Apriori method are identified. In real life business applications this can be used as a case where the company wants to sell multiple products together as a package and they can choose the itemset like {p, q, r} which will be become frequent over the time based on their business strategy. Selling multiple products at a time is always a subject of interest for any business organization and this concept will help them in their business planning. We are going to address the issues of missing itemset of Apriori Algorithm that are potentially the subject of interest and have huge business interest apart from making it faster.

Furthermore in traditional Apriori to check whether an k-itemset is valuable or not all the iterative steps starting from computing a single frequent item up to the k-itemset should be completed and that results in the exponential number of computations [2]. Henceforth decreasing the computations to identify business-critical frequent patterns is another challenge. We also address this issue here.

In this work we have two unique objectives – (i) to identify the infrequent itemset, that has certain hidden business critical patterns but is generally pruned by the traditional Apriori method (ii) to reduce the computational complexity to identify the frequent patterns.

4 Methodology

In this work two unique algorithms are proposed to – (i) identify the hidden business critical patterns that are otherwise pruned by the traditional Apriori method (ii) reduce the computational complexity to identify the important patterns. The detailed approach is explained in the next subsections.

4.1 Identifying the Hidden Business Critical Patterns

In this work, a novel strategy is proposed to identify these business critical infrequent itemsets that would be otherwise pruned if the traditional pruning strategy is followed. A modified Apriori algorithm (Algorithm 1) to generate the interesting patterns is proposed and explained here.

Algorithm 1: Algorithm Generate Interesting Patterns
Input: Transactional database, minimum threshold €
Output: All interesting items
Begin
1. Store all the frequent single items in L_1
2. $k \leftarrow 2$
3. While $L_{k-1} \neq \emptyset$
   a. Compute all possible k pair combinations of items in L_{k-1} and store in set C_k

      //Unlike traditional Apriori method, instead of scanning the entire database, the minimum support count of //immediate subsets is compared with the threshold to determine the interesting patterns
   b. For each item p in C_k
      i. Compute minimum support of all subsets of p and store in s
      ii. If $s \geq €$
          $L_k \leftarrow s$
   c. $k = k + 1$
4. return $\bigcup_k L_k$

End

The algorithm (Algorithm 1) begins with identifying all frequent single items by comparing them with the given threshold support count €. After eliminating all the infrequent items the rest are stored in L_1. After that k-item set is found by combining all the k-1 items or itemsets with each other. Thereafter minimum support count of all possible subsets of each of the k-pair itemset is computed. This support count is compared with the threshold support count € and if it is greater than or equal to € then the corresponding itemset is included otherwise it is pruned. Some future potential business-critical items that are pruned in the traditional Apriori process are preserved here as demonstrated by the Example 2.

In Table 1 and Table 2 the support count of 4-pair itemset {A, B, C, D} and support counts of all its possible subsets are shown. The threshold support count is assumed to be 2.

Example 2:

Table 1: Support count of 4-pair item set

Itemset	Support Count
{A, B, C, D}	1

In Table 1 it is observed that {A, B, C, D} has a support count 1 which is less than the threshold support count 2 whereas support counts of all the 3-pair item sub sets are greater than the threshold value. In the traditional Apriori pruning strategy {A, B, C, D} is pruned but in the proposed conditional pruning method since all the subsets are frequent, the superset is considered as an important itemset. The novelty of this study lies here in discovering hidden patterns that will add to the business insight and market forecast.

Table 2: Support counts of 3-pair item sets

Item set	Support Count
ABC	2
ACD	3
ABD	2
BCD	4

4.2 Reduction of the Computational Complexity for Frequent Pattern Identification

In the traditional Apriori method to identify whether an itemset of size k is frequent or not involves a large number of iterative stages. All possible combination of items (except the pruned items) and their support counts are computed until the given itemset is found and it involves nearly n*(k-1) database scans where n is the total number of computed items or itemsets (n≅2^(k-1)) [2]. Therefore these repetitive database scans and large number of computations make the overall Apriori method significantly costly. A unique top-down approach is proposed here to significantly reduce the computational complexity of interesting pattern identification. First frequent single items are found and thereafter all possible two pair itemsets along with their support counts are computed. Here a no pruning strategy of 2-pair itemsets is taken for faster operation. The results are internally stored for later use. These two pair itemsets are periodically refreshed to have the updated items. After computing and storing the interesting one item and two item pairs the next task is to check whether a larger itemset is frequent or important to the business. The decision that whether a given pattern is important for the business or not is taken by observing the support counts of all possible two pair subsets of the larger set. If the support counts of all of the two pair items are greater than the threshold then the itemset is considered valuable. The overall algorithm(Algorithm 2) is described below.

Algorithm 2: **Algorithm Compute Frequent Single And Two Pair Items**
Input: Source transactional database D and minimum support €.
Output: True if the given item set is frequent otherwise false.
Begin
1. Declare is Important = true
2. Search for all single items whose support count is more than the threshold count €
3. Compute all two pair items and store them in a list L.
4. For item set I in L
   a. Compute the frequency of I in the source database D
   b. Store I and corresponding support count

End for

5. Store the items in the given pattern in a set S
6. Compute all possible two pair subsets of set S and store them in Q
7. For each subset s in Q
   a. Get the support count of s from the correspond-ing stored database value and store in s
   b. s ≥ €
      continue
   else
      is Important = false
      exit loop and go to step 5.

End for

8. print is Important

End

A sample source transactional dataset is shown in Table 3. The frequent single itemsets generated by scanning the transaction dataset considering a threshold of two is depicted in Table 4. Again Table 5 depicts the set of all 2-itemsets.

Table 3: Transactional database

Transactions	Items
T1	A, B
T2	B, C, D
T3	A, B
T4	A, C
T5	A, C, D
T6	B, C

Table 4: Frequent single itemsets

Item	**Support count**
A	**4**
B	**4**
C	**3**
D	**2**

Table 5: Two itemsets

Item Sets	Support Count
AB	2
AC	2
BC	2
AD	1
BD	1
CD	2

Now to check a 3-item set {A, B, D} is frequent or not all possible two pair subsets and their support counts are obtained from Table 5. Here the sub sets are {A,B}, {B,D} and {A,D}. Among all these two pair itemsets BD is infrequent since its support count is less than the threshold support count 2. Hence ABD is considered as unimportant itemset. As an another example if itemset {A, B, C} is considered, then all of its two pair subsets are frequent and hence it is considered as an interesting pattern.

5 Results and Discussion

In the traditional Apriori algorithm to check whether an n itemset is important or not, the iterative steps of the Apriori methods have to repeat to obtain all n pair frequent itemsets. Thereafter the given item is searched among those itemsets to know whether it is frequent or not. In this study, a novel methodology is proposed where the traditional Apriori is followed to obtain two itemsets. Thereafter the decision that whether a given itemset is interesting or not is taken by comparing the support counts of all the subsets with the threshold support count. Since only two initial iterations of the traditional Apriori method is performed here, hence this method is computationally faster than the classic Apriori method. The number of computations of Apriori algorithm is exponential and nearly equal to 2^n where n is the number of items. In the proposed methodology since only two pair items are computed the number of computations is reduced to ${}^nC_2$ i.e. Table 6 gives a comparative study with respect to number of computations in both the methods while Figure 3 graphically represents the study.

Table 6: Showing number of computations in both approaches

Number of items	Number of computations in traditional Apriori	Number of computations in the proposed methodology
3	8	3
4	16	6
5	32	10
6	64	15
7	128	21
8	256	28
9	512	36

Apart from reduction in time complexity, mining the interesting patterns that are otherwise unexplored in traditional Apriori is another feature of this study. As an example in Table 7 a sample retail dataset is shown.

Now the single and two pair frequent items are shown in Table 8 and Table 9 considering threshold support count as 2.

From Table 9 frequent 3-itemsets are computed according to traditional Apriori algorithm as shown in Table 10.

According to traditional Apriori only the set { Bread, Butter, Jam } is considered as frequent or as important itemset as its support count is greater than the minimum support count. All other three itemsets (showed in the dashed rectangular area – Table 10) are discarded.

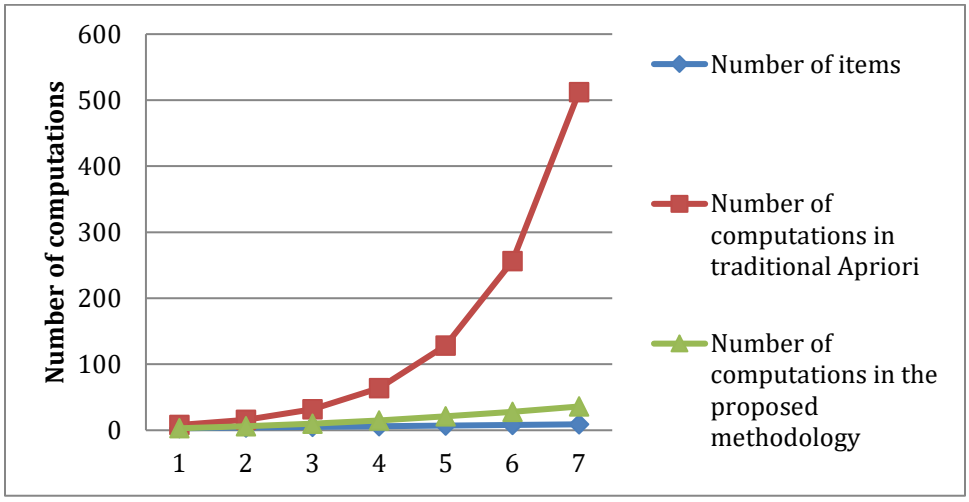

Figure 3: Comparative study of number of computations between traditional Apriori and the proposed method

Table 7: Retail dataset

Transactions	Item pattern
T1	Bread, Butter, Jam
T2	Bread, Jam
T3	Milk, Bread
T4	Milk, Bread, Butter
T5	Bread
T6	Butter, Milk
T7	Bread, Butter, Jam

Table 8: Frequent single itemsets

Items	Support count
Bread	6
Butter	4
Milk	3
Jam	3

Table 9: Frequent 2-itemsets

Item pair	Support count
Bread, Butter	3
Bread, Jam	3
Milk, Bread	2
Milk, Butter	2
Butter, Jam	2
Milk, Jam	1

Table 10: Frequent 3-itemsets

Item set	Support count
Bread, Butter, Jam	2
Bread, Milk, Jam	0
Butter, Milk, Jam	0
Milk, Bread, Butter	1

In this study after computing the frequent two itemsets the repetitive steps of Apriori are not continued, instead a top down approach is taken to mine the interesting patterns. As mentioned in Section 3, a particular itemset is considered as interesting if all the subsets are frequent. Here source database is not scanned again to find the frequency of the pattern. The item superset is considered as valuable if the minimum support count of all the subsets is greater than the threshold support count. The corresponding computations are shown in Table 11.

Table 11: Mining of interesting patterns in the proposed approach

Itemset	Subsets	Min subset support count	Selected / Discarded
{Bread, Butter, Jam}	{Bread, Butter}, {Bread, Jam}, {Butter, Jam}	2	Selected
{Bread, Milk, Jam}	{Bread,Milk}, {Bread,Jam}, {Milk, Jam}	1	Discarded
{Butter, Milk, Jam}	{Butter,Milk}, {Bread,Jam},{ Milk, Jam}	1	Discarded
{Milk,Bread, Butter}	**{Milk, Bread}, {Milk, Butter}, {Bread, Butter}**	**2**	**Selected**

In the above analysis it is observed that in traditional Apriori the combination {Milk, Bread, Butter} is ignored whereas in the proposed methodology this is considered as a potential interesting pattern. Hence for the retail market the combination **{ Milk, Bread, Butter** } can be tried to check

whether it will add value to the business or not. To check if a given itemset of any length is frequent or not, all possible two pair subsets are computed and the minimum support count of the subsets is compared with the threshold support to identify whether it is valuable or not.

6 Conclusion

Apriori algorithm is widely used by different business applications to identify the frequent itemset. But it misses some interesting business patterns. Here we identify these patterns to generate the itemset that can be useful for the organization to sell multiple products as a package. This will help the organization to increase the sales of the products which are below the threshold level. As the users get the chance to use these additional products at a little extra cost (the package of products are sold at discounted price) they may like it and may purchase that product individually also. Without spending a huge amount for the advertisement, the sales of the product will get a boost using our proposed approach. We also address another major drawback of the Apriori algorithm that is the higher number of computations. Our proposed methodology based on 2 itemsets drastically reduces the computational time. In Section 5 we have explained the computational benefit of our proposed method over Apriori. Henceforth the proposed approach will help the organization to identify the missing business pattern and to develop the business strategy based on these additional itemsets at a fraction of time over Apriori. The proposed method can be experimented over different big data tools for faster execution time. Further improvement is possible by incorporating parallelism in computation process may be introduced by using the concept of Mapreduce in Hadoop framework or Sharding in MongoDB.

References

[1] S. A. Abaya, "Association Rule Mining Based on Apriori Algorithm in Minimizing Candidate Generation," *International Journal of Scientific & Engineering Research*, 3(7):1-4, 2012.

[2] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules," *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, 1215:487-499, 1994.

[3] M. Al-Maolegi and B. Arkok, "An Improved Apriori Algorithm for Association Rules," *arXiv preprint arXiv:1403.3948,* 2014.

[4] R. Chang and Z. Liu, "An Improved Apriori Algorithm," *Proceedings of 2011 International Conference on Electronics and Optoelectronics*, IEEE, 1:V1-476, 2011.

[5] U. Fayyad, G. Piatetsky-Shapiro and P. Smyth, "From Data Mining to Knowledge Discovery in Databases," *AI Magazine,* 17(3):37-37, 1996.

[6] M. Giridhar, S. Sen, and A. Sarkar, "Share Market Sectoral Indices Movement Forecast with Lagged Correlation and Association Rule Mining," *IFIP International Conference on Computer Information Systems and Industrial Management*, Springer, Cham, pp. 327-340, 2017.

[7] Y. Guo, M. Wang and X. Li, "Application of an Improved Apriori Algorithm in a Mobile E-Commerce Recommendation System," *Industrial Management & Data Systems*, pp. 287-493, 2017.

[8] E. H. Han, G. Karypis and V. Kumar, "Scalable Parallel Data Mining for Association Rules," *AcmSigmod Record,* 26(2):277-288, 1997.

[9] J. Han, H. Cheng and D. Xin, "Frequent Pattern Mining: Current Status and Future Directions," *Data Mining and Knowledge Discovery,* 15(1):55-86, 2007.

[10] J. Han, J. Pei and Y. Yin, "Mining Frequent Patterns without Candidate Generation," *ACM Sigmod,* 29(22)1-12, 2000.

[11] J. L. Lin and M. H. Dunham, "Mining Association Rules: Anti-Skew Algorithms," *Proceedings 14th International Conference on Data Engineering*, IEEE, pp. 486-493, 1998.

[12] G. Maji, S. Sen and A. Sarkar, "Business Intelligence Development by Analysing Customer Sentiment," *2018 7th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, IEEE, pp. 287-290, 2018.

[13] S. Paladhi, S. Chatterjee, T. Goto and S. Sen, "AFARTICA: A Frequent Item-set Mining Method Using Artificial Cell Division Algorithm," *IGI Global Journal of Database Management*, 30(3):71-93, 2019.

[14] R. Perego, S. Orlando and P. Palmerini, "Enhancing the Apriori Algorithm for Frequent Set Counting," *International Conference on Data Warehousing and Knowledge Discovery*, Springer, Berlin, Heidelberg, pp. 71-82, 2001.

[15] S. Rao and P.Gupta, "Implementing Improved Algorithm over Apriori Data Mining Association Rule Algorithm 1," *Citeseer*, pp. 489-493, 2012.

[16] A. Savasere, E. R. Omiecinski, and S. B. Navathe, *An Efficient Algorithm for Mining Association Rules in Large Databases*, Georgia Institute of Technology, 1995.

[17] J. Singh, H. Ram and D. J. S. Sodhi, "Improving Efficiency of Apriori Algorithm using Transaction Reduction," *International Journal of Scientific and Research Publications*, 3(1):1-4, 2013.

[18] H. Toivonen, "Sampling Large Databases for Association Rules," 96:134-145, 1996.

[19] W. Yu; X. Wang, F. Wang, E. Wang and B. Chen, "The Research of Improved Apriori Algorithm for Mining Association Rules," *2007 International Conference on Service Systems and Service Management*, IEEE, pp. 1-4, 2007.

Anjan Dutta has completed his Bachelor of Technology in Information Technology from RCC Institute of Information Technology, Kolkata, India and Master of Technology in Information Technology from Calcutta University, Kolkata, India. Currently, he is an Assistant Professor in the Department of Information Technology at Techno International New Town, Kolkata, India. His research area includes data mining, big data and machine learning.

Runa Ganguli completed her Master of Technology in Computer Engineering and Applications from A.K. Chowdhury School of IT, University of Calcutta, India in 2020, Masters of Science in Computer and Information Science from University of Calcutta, in 2014. She did her Bachelors of Science in Computer Science Honours from Asutosh College, University of Calcutta in 2012. She is currently working as an Assistant Professor in the Department of Computer Science, The Bhawanipur Education Society College, Kolkata, University of Calcutta, India since 2015. She has several papers in reputed peer reviewed journals and international conferences. Her main research interest includes Graph Database, Data mining and Social Network Analysis. She has 6 years of teaching experience.

Punyasha Chatterjee received the B.Tech., M.Tech. and Ph.D. degrees in Information Technology from the University of Calcutta, Kolkata, India, in 2003, 2005 and 2018 respectively. She is presently an Assistant Professor in the School of Mobile Computing and Communication, Jadavpur University, Kolkata, India, since 2012. Her area of interest includes adhoc networks, wireless sensor networks, Internet of Things and pervasive computing. She is a senior member of IEEE and member of ACM.

Narayan C. Debnath earned a Doctor of Science (D.Sc.) degree in Computer Science and also a Doctor of Philosophy (Ph.D.) degree in Physics. Dr. Narayan Debnath is currently the Founding Dean of the School of Computing and Information Technology at Eastern International University, Vietnam. He is also serving as the Head of the Department of Software Engineering at Eastern International University, Vietnam. Dr. Debnath has been the Director of the International Society for Computers and their Applications (ISCA) since 2014. Formerly, Dr. Debnath served as a Full Professor and Chairman of Computer Science at Winona State University, Minnesota, USA. Dr. Debnath has been an active member of the ACM, IEEE Computer Society, Arab Computer Society, and a senior member of the ISCA.

Soumya Sen has received a Ph.D. in Computer Science & Engineering from the University of Calcutta in 2016. He received his M.Tech in Computer Science & Engineering in 2007 and M.Sc. in Computer and Information Science in 2005 also from University of Calcutta. He has joined A. K. Choudhury School of Information Technology under University of Calcutta in 2009. Dr. Sen has around 90 research publications in International Journals and conferences. He has 3 international patents and also published 2 books. Dr. Sen is the TPC member of many conferences across the world and serves as reviewer for many International journals. He is a member of IEEE and ACM. Dr. Sen is a fellow of IETE (Institution of Electronics and Telecommunication Engineers). His current research area interests are Data Warehouse & OLAP Tools, Data Mining, and Distributed Database.