Apriori Based: Mining Positive and Negative Frequent Sequential Patterns

Vedant Rastogi
Assistant Professor, Department of Computer Science Engineering
Institute of Engineering & Technology, Alwar, Rajasthan, India

Vinay Kumar Khare
M.Tech Student, Department of Computer Science Engineering
Institute of Engineering & Technology, Alwar, Rajasthan, India

Abstract- Sequential pattern mining is aims to discover interesting sequential patterns in a sequence database and it is one of the essential data mining tasks widely used in various application fields. Positive and negative sequential Patterns mining is an aim to find more interesting sequential patterns, considering the different minimum support of each data item in a sequence database. Generally, the generation order of data elements is considered to find sequential patterns. Advancements in Statistics, Machine Learning, Artificial Intelligence, Pattern Recognition and Computation capabilities have evolved the present day's data mining applications and these applications have enriched the various fields of human life including business, education, medical, scientific etc.
Keywords – : Data Mining, Sequential Patterns, Data Elements.

I. INTRODUCTION

Data mining is the extraction of hidden predictive information from large databases, is a powerful new technology with great potential to help companies focus on the most important information in their data warehouses. The Apriori-based algorithms find frequent item sets based upon an iterative bottom-up approach to generate candidate item sets. Since the first proposal of association rules mining by R. Agrawal [1, 2], Nowadays, with the rapid development of information technology, especially the web service-based application, service-oriented architecture and cloud-computing, continually expanding data are integrated to generate useful information. Many techniques have been used for data mining. Association rules mining (ARM) is one of the most useful techniques. The challenges associated with ARM, especially for parallel and distributed data mining, include minimizing I/O, increasing processing speed and reducing communication cost [3]. A major concern in ARM today isto continues to improve algorithm performance.

1.1 DATA MINING

Data mining can be viewed as a result of the natural evolution of information technology. The database system industry has witnessed an evolutionary path in the development of the following functionalities data collection and database creation, data management (including data storage and retrieval, and database transaction processing), and advanced data analysis (involving data warehousing and data mining). For instance, the early development of data collection and database creation mechanisms served as a prerequisite for later development of effective mechanisms for data storage and retrieval, and query and transaction processing. With numerous database systems offering query and transaction processing as common practice, advanced data analysis has naturally become the next target [12].
It is the process of finding correlations or patterns among number of fields in large relational databases. It is primarily used today by companies with a strong consumer focus - retail, financial, communication, and marketing organizations. It enables these companies to determine relationships among "internal" factors such as price, product

positioning, or staff skills, and "external" factors such as economic indicators, competition, and customer demographics. It also enables them to determine the impact on sales, customer satisfaction, and corporate profits.

1.2 ASSOCIATION RULES

Association rules are statements that help uncover relationships between seemingly unrelated data in a relational database or other information repository. An example of an association rule would be "If a customer buys a dozen eggs, he is 80% likely to also purchase milk. Obtained from reference [6] "An association rule has two parts, an antecedent (if) and a consequent (then). An antecedent is an item found in the data. A consequent is an item that is found in combination with the antecedent. Association rules are created by analyzing data for frequent 'if/then' patterns and using the criteria support and confidence to identify the most important relationships. Support is an indication of how frequently the items appear in the database. Confidence indicates the number of times the if/then statements have been found to be true. In data mining, association rules are useful for analyzing and predicting customer behavior. They play an important part in shopping basket data analysis, product clustering, and catalog design and store layout.

Support is an indication of how frequently the items appear in the database. The support of an association rule is the percentage of groups that contain all of the items listed in that association rule. [4]

Support of item 'a' = (p/q)

p = Number of groups containing that item 'a'

q = Total number of groups

Confidence indicates the number of times the if/then statements have been found to be true. The confidence value indicates how reliable this rule is. The higher the value, the more often this set of items is associated together [6].

Confidence ('A' -> 'B') = Support ('A'U'B') / Support ('A')

Support ('A'U'B') = number of groups containing 'A' and 'B'.

Support ('A') = number of group containing 'A'.

In data mining, association rules are useful for analyzing and predicting customer behavior. They play an important part in shopping basket data analysis, product clustering, and catalog design and store layout.

1.3 SEQUENCE DATABSE

A sequence database consists of sequences of ordered elements or events, recorded with or without a concrete notion of time. There are many applications involving sequence data. Typical examples include customer shopping sequences, Web click streams, biological sequences, sequences of events in science and engineering, and in natural and social developments. So a sequence is represented as an ordered list of data elements. A sequence database can be represented as a tuple <SID, Sequence Item List>, where SID: represents the sequence identifier and Sequence Item List specifies the sequence. Here follows the example.

Table 1.1: sequence database

SID	SEQUENCES
10	<a(abc)(ac)d(cf)>
20	<(ad)c(bc)(ae)>
30	<(ef)(ab)(df)cb>

40	<eg(af)cbc>

1.4 SEQUENTAIL DATA

Sequential data referred to as temporal data, can be thought of as an extension of record data, where each record has a time associated with it [9]. Consider a retail transaction data set that also stores the time at which the transaction took place. This time information makes it possible to find patterns such as "candy sales peak before Halloween."A time can also be associated with each attribute. For example, each record could be the purchase history of a customer, with a listing of items purchased at different times. Using this information, it is possible to find patterns such as "People who buy DVD players tend to buy DVDs in the period immediately following the purchase". The below table is shows an example of sequential transaction data. There are five different times- t1, t2, t3, t4, and t5: three different customers—C1, C2, and C3: and five different items—A, B, C, D, and E [13].

Table 1.2: Sequential transaction data

Time	Customer	Items purchased
t1	C1	A, B
t2	C3	A, C
t2	C1	C, D
t3	C2	A, D
t4	C2	E
t5	C1	A, E

Customer	Time and Items Purchased
C1	(t1: A, B) (t2: C, D) (t5: A, E)
C2	(t3: A, D) (t4: E)
C3	(t2: A, C)

1.5 FINDING SEQUENTAIL PATTERNS

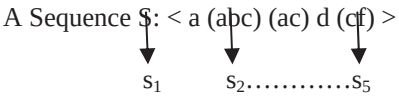
Terminology the length of a sequence is the number of item sets in the sequence. A sequence of length k is called a k-sequence. The sequence formed by the concatenation of two sequences x and y is denoted as x.y. The support for an item set i is defined as the fraction of customers who bought the items in i in a single transaction. Thus the itemset i and the l-sequence (i) have the same support. An item set with minimum support is called a large itemset or Litemset. Note that each itemset in a large sequence must have minimum support. Hence, any large sequence must be a list of Litemsets [9].

1.6 SEQUENTAIL PATTERN MINING

Sequential pattern mining is the mining of frequently occurring ordered events or Subsequences as patterns. An example of a sequential pattern is "Customers who buy a Canon digital camera are likely to buy an HP color printer within a month." For retail data, sequential patterns are useful for shelf placement and promotions. This industry, as well as telecommunications and other businesses, may also use sequential patterns for targeted marketing, customer retention, and many other tasks.

Other areas in which sequential patterns can be applied include Web access pattern analysis, weather prediction, production processes, and network intrusion detection. Notice that most studies of sequential pattern mining concentrate on categorical (or symbolic) patterns, whereas numerical curve analysis usually belongs to the scope of trend analysis and forecasting in statistical time-series analysis.[9]

Let I= {a, b, c, d, e, and f} be the set of all items. An item set is a nonempty set of items. A sequence is an ordered list of events. A sequence 'S' is denoted S=<s_1, s_2...s_l>, where event s_1 occurs before s_2, which occurs before s3, and so on.



Event s_j is also called an element of sequence S. In the case of customer purchase data, an event refers to a shopping trip in which a customer bought items at a certain store. The event is thus an item set, that is, an unordered list of items that the customer purchased during the trip. The item set (or event) is denoted s_j=<x_1, x_2… xk>where x_k is an item. For brevity, the brackets are omitted if an element has only one item, that is, element (x) is written as x. Suppose that a customer made several shopping trips to the store.
These ordered events form a sequence for the customer. That is, the customer first bought the items in s_1, and then later bought the items in s_2, and so on. An item can occur at most once in an event of a sequence, but can occur multiple times in different events of a sequence. The number of instances of items in a sequence is called the length of the sequence. A sequence with length l is called an l-sequence. A sequence α=< a_1 a_2 … a_n > is called a subsequence of another sequence β=< b_1 b_2 … b_m >and β is a super sequence of α, denoted as $\alpha \subseteq \beta$, if there exist integers $1 \leq j_1 < j_2 < \ldots < j_n \leq m$ such that $a_1 \subseteq b_{j1}$, $a_2 \subseteq b_{j2}$… $a_n \subseteq b_{jn}$.
For example, if α=< (ab), d> and β=< (abc), (de)> where a, b, c, d, and e are items, then α is a subsequence of β and β is a super sequence of α. A sequence database, S, is a set of tuples, <SID, Sequence Item List >, where SID is a sequence identifier and Sequence Item List is a sequence. For an example, S contains sequences for all customers of the store. A sequential pattern with length l is called an l-pattern.[16]

II. LITERATURE SURVEY

The problem of mining sequential patterns in the large databases introduced by Rakesh et al. [9] by presenting pre algorithms such as apriori sum, apriori all, albeit apriori sum.
Discovering patterns in sequence of events has been active area and can viewed in some literature by discovering the rule underlying the generation of given sequence in order to be able to predict a plausible sequence continuation and even interesting in finding all common patterns embedded in a database of sequences of sets events.

2.1 MINING SEQUENTAIL PATTERNS

The following five phases proposed by Rakesh et al. [9] were briefed.
1. Sort Phase. The database ***(D)*** is sorted, with customer-id **as** the major key and transaction-time **as** the minor key. This step implicitly converts the original transaction database into a database of customer sequences.
2. Litemset Phase In this phase Rakesh et al. [9] find the set of all Litemsets *L.* they are also simultaneously finding the set of all large l-sequences, since this set is just {(l) | l ϵ L}.
The problem of finding large item sets in a given set of customer transactions, albeit with a slightly different definition of support. The support for an itemset has been defined as the fraction of transactions in which an itemset is present, whereas in the sequential pattern finding problem, the support is the fraction of customers who bought the itemset in any one of their possibly many transactions. The main difference is that the support count should be

incremented only once per customer even if the customer buys the same set of items in two different transactions.[15]

Table 2.1: Customer-sequence database

Customer id	Customer sequence
1	{(30)(90)}
2	{(10,20)(30)(40,60,70)}
3	{(30,50,70)}
4	{(30)(40,70)(90)}
5	{(90)}

Table 2.2: Frequent positive sequential patterns

Sequential patterns with support>25%
{(30)(90)} {(30)(40,70)}

The set of Litemsets is mapped to a set of contiguous integers. In the example database given in Table 1.3, the large itemsets are (30), (40), (70), (40 70) and (90). The reason for this mapping is that by treating Litemsets as single entities, we can compare two Litemsets for equality in constant time, and reduce the time required to check if a sequence is contained in a customer sequence.

3. Transformation Phase. It is need to repeatedly determine which of a given set of large sequences are contained in a customer sequence. To make this test fast, we transform each customer sequence into an alternative representation.
In a transformed customer sequence, each transaction is replaced by the set of all itemsets contained in that transaction. If a transaction does not contain any Litemset, it is not retained in the transformed sequence. If a customer sequence does not contain any Litemset, this sequence is dropped from the transformed database. However, it still contributes to the count of total number of customers. A customer *se*quence is now represented by a list of sets of Litemsets. Each set of Litemsets is represented by $\{I_1, I_2, I_3 \ldots ..In\}$, where I_i is a Litemset.
This transformed database is called D_T. Depending on the disk availability, we can physically create this transformed database, or this transformation can be done on-the-fly, as we read each customer sequence during a pass. The transformation of the database in Fig. 2 is shown in Fig. *5.* For example, during the transformation of the customer sequence with Id 2, the transaction (10,20) is dropped because it does not contain any Litemset and the transaction (40 60 70) is replaced by the set of Litemsets {(40), (70), (40 70)).

4. Sequence Phase. Use the set of Litemsets to find the desired sequences.

5. Maximal Phase. Find the maximal sequences among the set of large sequences**,** this phase is combined with the sequence phase to reduce the time wasted in counting non maximal sequences. Having found the set of all large sequences *S* in the sequence phase, the following algorithm can be used for finding maximal sequences. Let the length of the longest sequence be n. Then,

For (k = n; k > 1; k--) do

For each k-sequence S_k does
De1ete from *S all* subsequences of S_k

2.2 EXTRACTION OF NEGATIVE ASSOCIATION RULES

As stated by Olena Daly et.al [8] about Exception Rules Mining Based on Negative Association Rules is explained as follows. Association rule is an implication of the form X=>Y, where X and Y are database
Itemsets. The example could be supermarket items purchased together frequently. Two measures have been developed to evaluate association rules, which are support and confidence. Association rules with high support and confidence are referred to as strong rules
Negative association rule is an implication of the form X=>~Y, ~X=>Y, ~X=>~Y, where X and Y are database itemsets, ~X, ~Y are negations of database items. Negative association rules consider both presence and absence of items in the database record and mine for negative implications between database items.[14]

2.2.1 EXCEPTION RULES

In association rules mining only the rules with high support and high confidence are considered as interesting rules. The generated patterns represent the common trends in the databases and are valuable for the marketing campaigns.
The rules with low support are just as valuable as they may contain unusual, unexpected knowledge about databases. Exception rules have been defined as rules with low support and high confidence. A traditional example of exception rules is the rule Champagne => Caviar. The rule may not have high support but it has high confidence. The items are expensive so they are not frequent in the database, but they are always brought together so the rule has high confidence. Exception rules provide valuable knowledge about database patterns.[10]
In association rules mining user-specified minimum confidence (minconf), minimum support (minsup) is given. Association rules with support>=minsup and confidence>=minconf are referred to as strong rules. Itemsets that have support at least equal to minsup are called frequent itemsets. Negative itemsets are itemsets that contain both items and their negations (for example XY~Z). ~Z means negation of the item Z (absence of the item Z in the database record).
Negative association rule is an implication of the form X=>~Y, ~X=>Y, ~X=>~Y, where X and Y are database items, ~X, ~Y are negations of database items. Examples of negative association rules could be Meat=>~Fish, which implies that when customers purchase meat at the supermarket they do not buy fish at the same time, or ~Sunny=>Windy, which means no sunshine implies wind, or ~OilStock=>~PetrolStock, which says if the price the oil shares is falling, petrol shares price will be falling too.
Exception Rules are rules with low support and high exceptionality values. In case of exception rules in association mining the confidence measure is not applicable to evaluate the exception rules. For example, we obtain a strong rule A=>B and would like to evaluate a potential exception rule A=>Not B. The strong rule A=>B has high confidence, implying that A=>Not B cannot have high confidence. Let us say the minimum confidence is 60%. The strong rule A=>B satisfies the minimum confidence constraint, so at least 60% of database records containing A also contain B. It means that maximum 40% records containing A do not contain B. The exception rule A=>Not B has maximum 40% confidence. As confidence is not applicable for evaluating exception rules, we propose a special measure exceptionality to evaluate the exceptions.

2.2.2 EXCEPTIONS IN NEGATIVE SENSE

As stated by Olena Daly et.al [8] that in order to obtain steady patterns of items occurred frequently is based on the mining the positive and negative association rules. Let us say X and Y are database items and

X

Are frequent XY is frequent …….. (1)
Y
X=>Y high confidence
Also we obtain that
X~Y
or — is infrequent …….(2)
~XY
So we have a strong association rule (1), and we make sure that (2) are infrequent. (1) and (2) are our premises to check if one of the rules (3) has a high exceptionality, which would prove it is an exception in negative sense.
X=>~Y
or — if high exceptionality then Exception (3)
~X=>Y

Example
Consider two oil companies X and Y. Their stock normally goes up at the same time: X=>Y In the case when their shares do not go up at the same time X=>~Y we call the rule X=>~Y an exception if X~Y is infrequent and has high exceptionality measure.[13]

2.2.3 EXCEPTIONS IN POSITIVE SENSE

As stated by Olena Daly et.al [8] that in order to obtain steady patterns of items occurred frequently is based on the mining the positive and negative association rules. Let us say X and Y are database items and
X
Are frequent X~Y is frequent …… (1)
Y
X=>~Y
Or — has high confidence……. (2)
~Y=>X
Also we obtain that
XY is infrequent (2)
We have a strong negative association rule (1), and we make sure that (2) is infrequent. (1) And (2) are our premises to check if one of the rules (3) has a high exceptionality, which would prove it is an exception rule in positive sense.
X=>Y
Or — if high exceptionality then Exception (3)
Y=>X

Example
Consider two oil companies X and Y. Their stock never goes up at the same time: X=>~Y In the case when their shares do go up at the same time X=>Y we call the rule X=>Y an exception if XY is infrequent and has high exceptionality measure.

2.2.4 ALGORITHM FOR MINING EXCEPTION RULES

Association rules are generated from frequent itemsets satisfying high confidence constraint. The confidence calculation is a straightforward procedure after all frequent
Itemsets have been generated. We do not consider the confidence calculation as it is easy and conceptually proven correct. The input of the exception rules mining algorithm are frequent 1-itemsets. The output of the algorithm is exceptional itemsets. Exceptional itemsets will become exception rules after the confidence of association rules has been checked. They generate frequent itemsets and on each step k (k is the length of the itemset). They check the

conditions (1), (2) from sections 2.2.2 and 2.2.3 and if they hold true, we check the exceptionality values for candidate exceptions.

2.3 SINGLE MINIMUM SUPPORT APPROACHES

In earlier data mining algorithms such as apriori, apriori sum used single minimum support to mine the frequent patterns by giving equal importance to each item occurred in database. The following are some algorithms where single minimum support is used.

2.3.1 APRIORI ALGORITHM

The Apriori algorithm employs an iterative level-wise search for generating frequent item sets. An item set is a set of items. A candidate k-item set refers to an item set having 'k' number of items and frequent k-item set refers to subset of candidate k-item sets whose support is greater than or equal to user specified *min sup*. The Apriori algorithm repeats the steps from (i) to (iii) starting with k=1 till no more frequent item sets are found [6]:

(i) C_k is generated.
(ii) L_k is generated from *Ck* by pruning the item sets whose support is greater than min sup value.
(iii) C_{k+1} are generated by joining L_k with itself.

2.4 MULTIPLE MINIMUM SUPPORT APPROACHES

To avoid the rare item problem and to improve the performance of extracting frequent itemsets involving rare items, an approach known as Multiple Minimum Support approach.

2.4.1 RARE ITEM PROBLEM

In many applications, some items appear frequently in the data, while others appear rarely. If the frequencies of items vary a great deal, we will encounter the dilemma called as the rare item problem. If minsup is set too high, we will not find those sequential patterns that involve rare items in the data. If minsup is set too low in order to find sequential patterns that involve both frequent and rare items, it will cause combinatorial explosion because those frequent items will be associated with one another in all possible ways and many of them are meaningless. In order to deal with the rare item problem, we put forward a multiple minimum supports based model for mining sequential pattern. In this model, we can specify a different minimum item support for each item.
To avoid the rare item problem there are so many algorithms used the multiple minimum support approach some of them are explained below.

2.4.2 MS APRIORI

In this approach, each item is assigned with a minsup value known as "Minimum Item Support" (MIS) and frequent itemsets are generated if an itemset satisfies the lowest MIS value among the respective items. The MIS value is assigned to each item equal to a percentage of its support. For every item $i_j \in$ I, the MIS (i_j) is calculated as per below equation. [3]

$$MIS\ (i_j) = \beta S\ (i_j)),\ if\ \beta S\ (i_j) > LS$$
$$= LS\ else$$

where, β is a user-specified proportional value which can be varied between 0 to 1, $S(i_j)$ refers to support of an item equal to $f(i_j)/N$, ($f(i_j)$ represents frequency of i_j and N is the number of transactions in a transaction dataset) and LS corresponds to user-specified least support value.

MSApriori suffers from the following problems.

1) If β is set high, it can be observed that MIS for rare items will be relatively more close (almost equivalent) to their supports as compared with frequent items. As a result, itemsets containing rare items fail to satisfy the support of (lowest MIS) rare item in that itemset. So, frequent itemsets involving rare items are missed.

2) To facilitate participation of rare items, MIS for the rare items have to be less than their support values. It can be achieved by setting low β value. However, this may cause frequent items to set very low MIS values. With low MIS values for frequent items, the items will be associating with one another in all possible ways, thereby generating.

2.4.2 RELATIVE SUPPORT APRIORI ALGIRITHM (RSAA)

Relative Support Apriori Algorithm (RSAA) has been proposed [5] for discovering frequent itemsets involving both frequent and rare items. In discovering frequent itemsets, RSAA uses three user specified measures: first support s_1, second support s_2 ($s_1 > s_2$) and relative support R_{sup}. Items having support greater than or equal to s_1 are considered frequent items and the items having support less than s_1 and greater than or equal to s_2 are considered as rare items. If an itemset contains only frequent items, it has to satisfy s_1 to be a frequent itemset. If an itemset contains rare items, its support has to satisfy s_2 and its R_{sup} should satisfy user specified minimum relative support mR_{sup} to be a frequent itemset. The main issue in this algorithm is providing values to these parameters for the given dataset. [3]

2.4.3 SUPPORT DIFFERENCE APPROACHES

Support difference (SD) refers to the acceptable deviation of an item from its frequency (or support) so that an itemset involving that item can be considered as a frequent itemset. For each item 'i_j ', calculation of minsup known as minimum item support (MIS (i_j)) is as follows:

$$\text{MIS } (i_j) = S (i_j) - SD \text{ when } (S (i_j) - SD) > LS$$
$$= LS \text{ otherwise}$$

Where, S (i_j) refers to support of item 'i_j' and LS refer to the user-specified least support. Using SD, MIS for the items range from ($-\infty$, $+\infty$). To prevent MIS values of the items in reaching 0 or lower, we use concept of least support (LS). Least support refers the lowest minimum support an item or itemset should satisfy to become a frequent itemset. The LS takes a value in the range [0%, 100%]. For a given dataset, the value of SD can be calculated as per below equation.

$$SD = \lambda (1 - \alpha)$$

Where λ represents the parameter like mean, median, mode, maximum support of the item supports and α is the parameter ranging between 0 and 1. SD takes values from (0, λ).

1) If $\alpha = 0$ then SD = λ. Higher the SD value, minimum supports for the items will be relatively less than their corresponding supports.

2) If $\alpha = 1$ then SD = 0. When SD = 0, minimum support for an item is equivalent to their corresponding support values.

REFERENCES

[1] http://www.anderson.ucla.edu/faculty/jason.frand/teacher/technologies/palace/datamining.htm.

[2] http://searchbusinessanalytics.techtarget.com/definition/association-rules-in-data.htm

[3] Agrawal R,Srikant R. "**Mining sequential patterns**" In the Proc.1995IntConf.onDataEngineering,Taibei,Taiwan,March1995

[4] Pang-Ning Tan, Michael Steinbach, Vipin Kumar [2006]. "**Introduction to Data Mining**" 3rd Edition. Pearson Education Inc.

[5] Weimin Ouyang, Qinhua Huang,"**Mining Positive and Negative Sequential Patterns with Multiple Minimum Supports in Large Transaction Databases",** IEEE, Second WRI Global Congress on Intelligent Systems 2010.

[6] Daly O, Taniar D**.," Exception Rules Mining Based on Negative association Rules**", Lecture Notes in Computer Science, Vol.3046, 2004, pp543-552**.**

[7] Savasere A, Omiecinski E and Navathe S. **Mining for Strong Negative Associations in a Large Database of Customer Transactions.** In Proc. 1998 Int. Conf. on Data Engineering, pp.494-502.
[8] Yun, H., Ha, D., Hwang, B., and Ryu K. H. "**Mining association rules on significant rare data using relative support.**", The Journal of Systems and Software 67, 2003, pp. 181-191.
[9] Liu, B., Hsu, W., and Ma, Y. "**Mining Association Rules with Multiple Minimum Supports.**" SIGKDD Explorations, 1999.
[10] R. Uday Kiran and P. Krishna Reddy," **An Improved Multiple Minimum Support Based Approach to Mine Rare Association Rules",** IEEE**,** 2009.
[11] Sue-Chen Hsueh1, Ming-Yen Lin2, Chien-Liang Chen2, "**Mining Negative Sequential Patterns for E-Commerce Recommendations",** IEEE, Asia-Pacific Services Computing Conference**,** 2008.
[12] Piatetsky-Shapiro G. **Discovery, "Analysis and Presentation os Strong Rules, Knowledge Discovery in Databases",** AAAI/MIT, Menlo Park, CA, 1991, pp.229-248.
[13] A. Marascu and F. Masseglia, "**Mining Sequential Patterns from Temporal Streaming Data" ,** In MSTD'05.
[14] A. Metwally, D. Agarwal, and A.E. Abbadi, " **Efficient Computation of Frequency and Top-k Elements in Data Streams",** In ICDT'05.
[15] J. Han, G. Dong, Y. Yin, **"Efficient Mining of Partial Periodic Patterns in Time Series Database"**, Proceedings of Fifth International Conference on Data Engineering, Sydney, Australia, IEEE Computer Society, 1999, pp.106-115.
[16] J. S. Park, M. S. Chen, P. S. Yu, **"An Effective Hash Based Algorithm for Mining association rule"**, Proceeding of the ACM SIGMOD Conference on management of data, 1995, pp. 175-186.