IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 18, Issue 2, Ver. V (Mar-Apr. 2016), PP 52-59
www.iosrjournals.org

Mining Frequent Patterns on Object-Relational Data

Juliana Lucas Mbuke[1], Lu Songfeng[2]
[1](*Department of ICT, College of Engineering and Technology, Mbeya University of Science and Technology, Tanzania)*
[2](*School of Computer Science and Technology, Huazhong University of Science and Technology, P.R China)*

Abstract : *Data mining is viewed as an essential part of the process towards knowledge discovery. Through data mining process different kinds of patterns that is frequent pattern and others, are discovered, evaluated and presented as knowledge. Mining data from large database repositories which contains vast amount of data is not only interesting but also essential since it yields useful and novel information which aid organizations and other individuals in decision making. Many previous research works based on frequent pattern mining algorithm and association rule mining are focused on transactional data, yet there are other interesting types of data which requires just about same study in order to perform mining techniques on them so as to discover novel information. This paper elaborates mining of Object-relational data in relation to transactional data as a base of understanding, later uses differed mining algorithms to uncover frequent patterns and evaluate performances of these algorithms. Two approaches for this mining task was proposed, namely fundamental approach and nested-relations approach.*

Keywords - *data mining, frequent pattern mining, object-relational data (ORD)*

I. Introduction

Data Mining (DM) has attracted more attention in recent years probably because of the popularity of the concept of "Big data". Simply DM can be referred in different manners for instance as an extraction of knowledge from an immense amount of data available in different data repositories primary databases and other data repositories such as data warehouses and Internet-based global information systems i.e. World Wide Web. It's also viewed as an essential part of the process towards Knowledge Discovery from Data (KDD) [1-3].

From a DM process different kinds of patterns such as frequent, infrequent, rare and negative patterns are discovered, evaluated and presented as novel knowledge that's essential. Information and knowledge discovered from DM process can be used for applications ranging from market analysis, fraud detection, and customer retention, to production control, web search, digital libraries, science exploration and discovery [4, 5]. KDD process generally consists of three phases namely *Pre-processing* (pre-processing data before mining), *Data mining*, *Post-processing* (pattern evaluation and knowledge presentation after data mining) [6].

1.1 Research incentive

This research was driven from shortcomings determined during analysis of most common and new algorithms used for frequent pattern mining. Based on the analysis, many frequent patterns mining algorithms architecture often operates on transactional and/or relational data model. However due to advances of technology in database systems many organizations have been adopting and acquire their Database Management System (DBMS) in ORD model, and this brings the need for further study of these instances and determine how to mine knowledge from them.

1.2 Study contribution

Research findings yield basic understanding of an ORD model, in form of their structure and operation in their Object Relational Database Management System (ORDBMS). Furthermore, it provided its perspective on how to mine frequent patterns on ORD using present frequent pattern mining algorithms, and later to evaluate performance of these algorithms in terms of run time and memory usage.

This research presents fundamental knowledge to organizations and users on how mining frequent patterns is vital as it enable one to be able find associations and correlation relationships in their ORD and be able to make decisions based on the findings. A well performed algorithm may be applied on organizations working object relational databases so as to determine frequent patterns on their data and be able to explore relationships between them.

II. Literature Review

"Pattern mining" is a data mining task that involves finding existing patterns in data [7]. FP mining is the mining of patterns which occur frequently in a certain targeted data set from data rich repositories, for

instance relational databases, data warehouses, transactional databases, object-relational databases and others.

It discovers implicit, previously unknown, and potentially useful knowledge in the form of patterns revealing frequently co-occurring items, events, or objects [8]. There are many kinds of frequent patterns mainly are itemset, sequential and structured patterns that appear in a data set with frequency no less than a user-specified threshold [5, 9-11].

Finding frequent patterns play an essential role in mining tasks that try to uncover interesting patterns among big data, such as association rules and correlations. Furthermore, it helps in other data mining tasks such as classification, and clustering and many more of which the mining of association rule is one of the most popular research area, and thus frequent pattern mining has become an important data mining task [12-15]. This section details evolution of data models and different mining algorithms.

2.1 Data models

One of the most important applications for computers is storing and managing information. The manner in which information is organized can have a profound effect on how easy it is to access and manage. Perhaps the simplest but most versatile way to organize information is to store it in tables. The relational model is centered on this idea as Codd in late 1960's to 1970, proposed that database systems should present the user with a view of data organized as two-dimensional tables called *relations* [16-18] .

Relational model can be applied to many different applications [19]. Moreover, its simplicity, the separation of logical from physical level and the ease of expressing operations declaratively by the Structured Query Language (SQL) led to wide adaptation of relational model by DBMS developers and DB application developers. Although by 1990, relational database systems were standard, yet the database field continues to evolve, and new issues and approaches to the management of data surface regularly. By then object-oriented features started to infiltrate the relational model [20].

In object-oriented data model main construct is an object. As in ER model we have entities and in relational model, there are relations similarly we have objects in OO data modeling [21]. In this model concept, any physical or abstract thing is regarded as objects and relates them to the real world. It can be a person, place, thing, or a concept. An object can be used to model the overall structure not just a part of it. Relational model has been extended to object-relational model by incorporation of features such as *structured types of attributes, methods, identifiers for tuples and reference*, adopt from OO model [20].

2.2 Mining algorithms

This section briefly describes some of the algorithms for mining frequent patterns. The main operation of these mining algorithms is to compute the occurrence frequency of the interesting set of items among a certain dataset [22]. The algorithms are of different categories such as vertical data format and horizontal data format [23-25].

- Horizontal data format includes algorithms such as Apriori algorithm, FP-growth algorithm and RARM algorithm
- Vertical data format include algorithm such as ECLAT algorithm.

Other algorithms include RElim.

2.2.1 Apriori algorithm

The algorithm was first proposed for mining frequent itemsets for finding association rules; further the algorithm uses *prior knowledge* of frequent itemset properties [26, 27]. An algorithm employs an iterative approach known as a *level-wise* search, such that large Itemset generated at *k* level are used to generate candidates at *k+1* level and the database is scanned multiple times as long as large frequent Itemsets are generated [5, 13, 14, 28-30]. To start, the set of frequent 1-itemsets is found by scanning the database to accumulate the count for each item, and collecting those items that satisfy minimum support count (*min_sup*) [31]. Then, the resulting set is denoted as L_1. Next, L_1 is used to find L_2, the set of frequent 2-itemsets, which is used to find L_3, and so on, until no more frequent *k*-itemsets can be found. The finding of each *Lk* requires one full scan of the database.

Even though the algorithm is said to popular and the earliest to be implemented for mining frequent patterns, there are two limitations to this same algorithm that's complex candidate itemset generation process which consumes large memory and enormous execution time, as well as excessive database scans for candidate generation [28].

2.2.2 FP- growth algorithm

This is the mining of frequent patterns without a candidate generation as compared to the Apriori algorithm detailed above. The algorithm uses a tree-based data structure, FP-tree, and the corresponding mining algorithm, FP-growth, for discovering frequent patterns [30, 32, 33]. The algorithm requires only two database

scans to complete the mining task. FP-growth algorithm shows that it is efficient and scalable for mining both long and short frequent patterns, and is about an order of magnitude faster than the Apriori algorithm.

2.2.3 RARM algorithm

It's an algorithm that avoids complex candidate generation process. By using a novel tree data structure known as *Support-Ordered Trie Itemset* (SOTrieIT), RARM accelerates the mining process even at low *support* thresholds [28, 34]. A *TrieIT* is a set of nodes in tree consisting of 2 values (Item Label and Item Support). *SOTrieIT* is a sorted ordered *TrieIT* in which nodes are sorted with their respective *support* count. Highest support item moves to the left most nodes and lowest support node is at the right most position in the tree.

As the main bottleneck in association rule mining and in Apriori algorithm is the generation of candidate 1-*Itemset* and 2-*Itemset*, Performance of RARM is radically improved by using *SOTrieIT*. Generating 1-*Itemset* and 2-*Itemset* is a time consuming process in data mining and candidate 1-*Itemset* and 2-*Itemset* can easily be extracted from the *SOTrieIT* [35].

2.2.4 ECLAT algorithm

Both Apriori and FP-growth methods mine frequent patterns from a set of transactions in *TID-itemset* format (that is, f*TID* : *itemsetg*), where *TID* is a transaction-id and *itemset* is the set of items bought in transaction *TID*, this data format is known as horizontal data format [13]. Alternatively, data can also be presented in *item-TID set* format (that is, f*item* : *TID set*g), where *item* is an item name, and *TID set* is the set of transaction identifiers containing the item. This format is known as vertical data format. ECLAT algorithm uses vertical database layout and the intersection based approach to compute the support of an itemset, but also it uses depth-first strategy to reduce size of generating collections of candidate itemsets [14, 36].

2.2.5 RElim algorithm

Recursive elimination is an algorithm for finding frequent item sets, which is strongly inspired by the FP-growth algorithm. It does its work without prefix trees or any other complicated data structures, processing the transactions directly. Its main strength is not its speed, but the simplicity of its structure [37].

III. Mining Frequent Patterns On ORD

This is the extraction of knowledge from object relational data by uncovering hidden frequent patterns from Object-relational databases (ORDBMS). In this section we proposed two approaches on how to mine ORD, in which one will be used in performance evaluation. The elaboration of this type of data to be mined will be done in comparison to/or following the transaction data platform, since most of frequent patterns mining algorithms were primary developed to find frequent itemset pattern on transactional databases.

Basically, ORD and transactional data have their similarities and their differences mainly because; ORD has some extension features of object-oriented data model. As detailed original relational data, could not store nested relations, but due to this extension features, ORD is able to store some nested relations and has many more other features to facilitate storage, modification and manipulation of DBMS.

3.1 Object-relational DBMS

ORDBMSs synthesize the features of Relational DBMSs with the best ideas of Object Oriented DBMSs. Although ORDBMSs reuse the relational model as SQL interprets it, the OR data model is opened up in novel ways. New data types and functions can be implemented using general-purpose languages such as C and Java. In other words, ORDBMSs allow developers to embed new classes of data objects into the relational data model abstraction.

With the operation of ORDBMS using ORD model, there several changes of structure of database architecture, though in a mining perspective, the only concern is the tuple identifier factor as it's a unique as the *object_ID*, and the nested relations which can now be implemented in a database as to compared with relational databases where nested relations could not be stored all at once, and later course to have many related tables in a relation storing the nested relations. Below on the next two subsections, we elaborate the proposed two approaches on how to mine ORD to obtain frequent patterns. These approaches were based on, with and without nested relation factors.

3.2 Fundamental Approach

This is the mining of ORD from object relational databases without nested relations. I used comparison of this structure of object relational data with that of the transactional data to bring out the similarity which is useful in the mining process of this approach. Let's compare this type of relation and that of transaction using sample data.

TABLE 1: Transaction data

Transaction_ID	*Item_IDs*
T01	I1, I2, I3
T02	I4, I5, I3
T03	I3, I2

TABLE 2: Object-relational data without nested relations

Object_ID	*Name*	*Address*	*Birthdate*	*Movie*
01	Fisher	Malibu	1999	Scandal
02	Hamil	B'wood	1988	Scandal
03	Fisher	B'wood	1888	Empire
04	Olive	B'wood	1988	Scandal

In a mining perspective, number of attributes in ORD is the number of items in transactional data, and number of instances equals to objects defined by *object_ID (OID)* in ORD and transactions defined by T_ID (*TID)* in transactional data.

The following are parameters needed to be defined during mining of ORD using different algorithms in comparison to transactional data, as it is well known that for frequent pattern mining tasks to take place two vital input parameter must be defined on the algorithm in order to output frequent patterns *The minimum support count and threshold*, using a support counts to define number of occurrence of items in a database, let's assume the minimum support count = 1 and total number of transactions or objects is 10, then to obtain a *min_sup* threshold,

The corresponding relative support is $\frac{\text{minimum support count}}{\text{Total number of transactions/objects}} = 0.1 \text{ or } 10\%$

Parameters for mining transactional data

Input: *D*, a database of transactions;
min sup, the minimum support count.
Process: Mining algorithm *(See Section 2)*
Output: *L*, frequent itemsets in *D*.

Parameters for mining Object Relational Database

Input: *D*, a database of objects;
min sup, the minimum support count.
Process: Mining algorithm *(See Section 2)*
Output: *L*, frequent attriSets in *D*.

In each algorithm for mining frequent patterns, it usually require at least a single scan of an entire database to count and determine number of occurrence each of item, a count which is compared to the minimum support threshold defined by the user to determine which items are worth taken as frequent items. Input parameters for all algorithms were; an input *arff file* that contains synthetic data for testing the algorithms, an output *text file* which define output structure, and a user defined minimum support threshold.
Output results in each algorithm include;

- *Apriori;* Candidate count and maximum candidate size, number of frequent itemsets, maximum memory usage in mb and a total time in ms
- *FP-growth;* Objects (Transactions) count, number of frequent itemsets, maximum memory usage and a total time
- *ECLAT;* Objects (Transactions) count, number of frequent itemsets, maximum memory usage and a total time
- *RElim;* Number of frequent itemsets, maximum memory usage and a total time

As it can be seen, using a fundamental approach, with an assumption made that the Object Relational Database is without nested relations; ORD can be mined in a quite similar manner as the current transactional databases. And for that reason, the mining algorithms won't need structure alteration to perform the data mining task of finding frequent patterns.

3.3 Nested-relations Approach

This is the proposed approach for mining ORD from their databases with nested relation, an extension feature from Object Oriented data model. Now, let consider only the Object relational dataset in comparison to transactional data in

Table 1.

TABLE 3: Nested relations on Object Relational data Stars with Nested relations address and movies

Object_ID	*Name*	*address*	*birthdate*	*Movies*
01	Fisher	*street* | *city*: Maple | H'wood; Locust | Malibu	1999	*title* | *year* | *length*: Star Wars | 1977 | 124; Scandal | 1980 | 127; Return | 1983 | 133
02	Hamil	*street* | *city*: Oak | B'wood	1988	*title* | *year* | *length*: Star Wars | 1977 | 124; Empire | 1980 | 127; Return | 1983 | 133

As the figures show that a database with nested relation requires a different approach towards mining of frequent patterns. Even though input and output parameters may be the same, a mining algorithm process will need an alteration before execution

Parameters for mining Object Relational Database with nested relations

Input: *D*, a database of objects;
min_ sup, the minimum support count threshold.

Process: Mining algorithm
First an algorithm needs to check for an existence of nested relations in an object
If exist nested relations
Then find frequent attriSets
Combine the current frequent attriSets with the remaining attriSets
Then, finds frequents attriSets for an entire database.

Note: this process may be different since each algorithm has its own structure.

Output: *L*, frequent attriSets in *D.*

This is a proposed approach for mining object relational data with nested relations. Even though the approach is in its ideal theoretical primary step, we are looking forward for more research to be conducted in order to actually implement this change on present algorithms.

IV. Results And Discussion

In this section, I present the performance result using the fundamental approach of mining object relational data. The performance results were obtained after mining frequent patterns using Apriori, FP-growth, ECLAT and RElim algorithms. These algorithms were executed in SPMF software version v0.96r16. All the performance evaluation of algorithm on mining ORD were done on LENOVO ThinkPad E445 computer of 2.10GHz AMD A8-5550M APU processor with 4GB RAM running windows 7 operating system of 64 bits.

To check the performance of these algorithms, I used Synthetic dataset available in the SPMF library. Below there is performance results of algorithms in a tabular form and their later evaluation in terms of run time, frequent patterns obtained and maximum memory acquired after execution of algorithms using range of minimum support threshold.

Note:

MinSup: 0.1 or 10% that is a minimum support threshold, based on dataset, experiments were conducted using the lowest *min_sup* of 0.1 and highest of 0.4.

Analysis of executed results are provided based on run time of an algorithm in milliseconds (ms) and their maximum memory usage in mega bits (mb) as the minimum support *(min_sup)* threshold varies. Comparison of performance evaluated is between all four algorithms.

4.1 Performance evaluation using *maximum memory* used against *min_sup*

TABLE 4: Tabular presentation of individual performance result of each algorithm on maximum memory usage against *min_sup*

(max mem mb vs min_sup)

	Apriori		FP-Growth		Eclat		Relim		
(min_sup)	max	mem *(mb)*	max	mem *(mb)*	max	mem *(mb)*	max	mem *(mb)*	*frequent count*
0.1	15		11.3		5.5		8.7		498
0.2	14		11		3.2		5.2		45
0.3	10.9		8.7		7.6		3.7		17
0.4	9.9		7.2		10.4		6.3		7

The tabular presentation of maximum memory usage against *min_sup* of Apriori algorithm and FP-growth algorithm depicts that; as the minimum support threshold increases, maximum memory usage decreases as well as the frequent patterns resulted. This is due to the rule of defined *min_sup*, that all attribute/items with less support count compared to the user defined minimum support count threshold are not considered as a frequent pattern.

However, ECLAT algorithm depicts inconsistence in maximum memory usage, as it keeps on increasing and decreasing throughout the experiment. As for RElim there is a small fluctuation at the memory usage where *min_sup = 0.4*, the memory changed drastically to increase, but it was on a good performance with the rest of the *min_sup* defined.

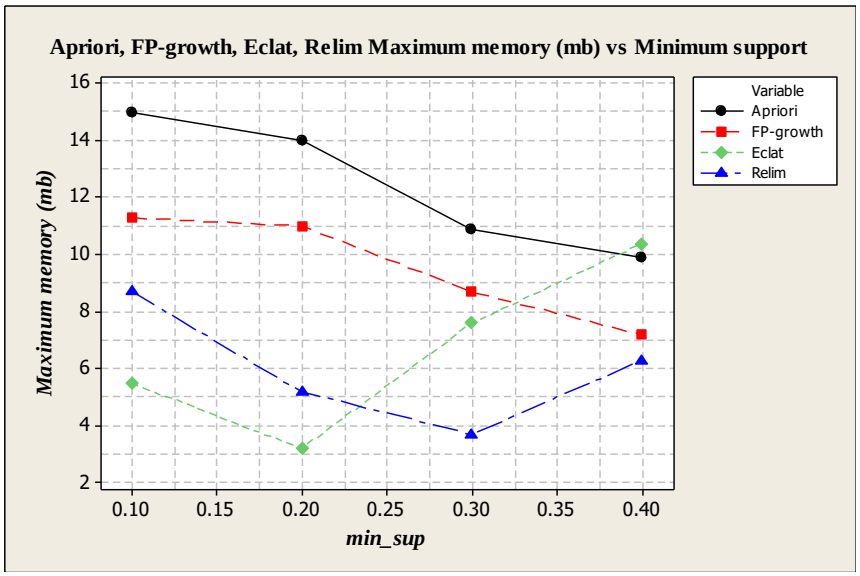

Fig.1. graphical presentation of performance result for all algorithms on maximum memory usage against *min_sup*

As the graphical presentation of all algorithms depicts that Apriori and FP-growth seem to be well utilizing the memory usage as the *min_sup* increase which led to lower number of resulted frequent patterns. Excluding the last *min-sup* of 0.4 defined, RElim out performs both Apriori and FP-growth in memory usage. Nevertheless, ECLAT algorithm had unsatisfactory performance as its results contain inconsistence.

4.2. Performance evaluation using *run time* used against *min_sup*

TABLE 2: Tabular presentation of individual performance result of each algorithm on run time against *min_sup*

(run time ms vs min_sup)

	Apriori	FP-Growth	Eclat	Relim	
(min_sup)	run time (ms)	run time (ms)	run time (ms)	run time (ms)	frequent count
0.1	64	55	58	36	498
0.2	7	5	6	11	45
0.3	10	2	8	7	17
0.4	9	1	7	8	7

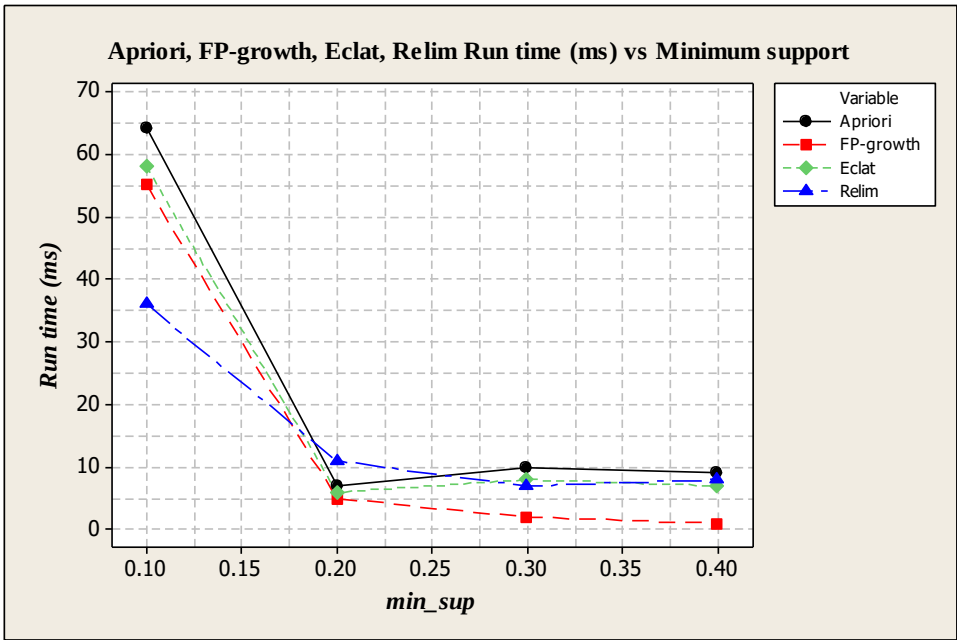

Fig.2. Graphical presentation of performance result for all algorithms on run time against *min_sup*

As both tabular and graphical presentation depicts, all algorithm are performing at satisfactory run time, as it can be seen that with the lowest *min_sup* of 0.1 run time were 64 high and 36 low, compared to the rest of the specified *min_sup* where the run time reduced as the *min_sup* defined increases, though FP-growth algorithm outperform rest of algorithms that is RElim, ECLAT and Apriori algorithm.

V. Conclusion

We studied Object-Relational Data (ORD) evolution, from relational data model towards an extension of relational data using object-oriented feature, which bring about an ORD. During the research period were able to examine different frequent pattern mining algorithms from the primal Apriori algorithm, FP-growth algorithm towards ECLAT algorithm and RElim algorithm.

Experiments for mining frequent patterns on ORD based on the fundamental proposed approach were done using an online data mining library named SPMF, were synthetic data were used to run different frequent pattern mining algorithms. Through the obtained results, we were able to evaluate performance of these algorithms, which was done based on run time, maximum memory usage and minimum support threshold parameters. Performance yield shows that among the run algorithms, RElim algorithm works better compared to Apriori, FP-growth and ECLAT algorithms in terms of memory usage against minimum support threshold, however in terms of maximum run time FP-growth algorithm shows it can perform better than the rest of algorithms.

Nevertheless, despite of this research finding shown on the performance evaluation results, still there is work to be done on particular two areas that are modification of frequent pattern algorithms and analysis of new data models.

- There is a need for further study and modification of these present algorithms to be able to run object relational data with nested relations, the second proposed approach. The alteration may begin on a primal algorithm that's Apriori up to recently frequent pattern (itemset) mining algorithms such as Post+ and FIN.
- Technology is becoming more and more advanced led to creation of new data models which acquire researchers to study and provide more efficiency algorithms for securely mining knowledge. Semi structured-data model including XML, and Post-relational data modeled by Conceptual Universal Database Language (CUDL) and Conceptual Universal Database Language Abstraction Level (CAL) are among the instances.

Acknowledgements

I would like to express mainly gratitude to Almighty God for His love, grace, protection and guidance throughout my studies.

My appreciation goes to my former supervisor Prof. Lu Songfeng for his guidance and cooperation as my teacher during my research, I am thankful for all that you have done during supervision. To my family especially my Mother I thank you for your prayers and being there for me, to my friends and former classmates

Lendon and others, we have been there for each other, your encouraging words, the time we spend right through our studies were precious, I am glad to have met you all. To my best friend Jared Zabron Massele, I cannot thank you enough for all you have done; your prayers, advice, assistance, persuasion and understanding throughout my studies and research have been very helpful.

References

[1]. Francisco Guil, A.B., and Roque Mar´ın, TSET: An Algorithm For Mining Frequent Temporal Patterns.
[2]. Li, L., B. Yang, and F. Zhou, A framework for object-oriented data mining, in Fuzzy Systems and Knowledge Discovery, 2008. FSKD'08. Fifth International Conference on. 2008, IEEE. p. 60-64.
[3]. Gole, S. and B. Tidke, Frequent itemset mining for Big Data in social media using ClustBigFIM algorithm, in Pervasive Computing (ICPC), 2015 International Conference on. 2015, IEEE. p. 1-6.
[4]. LEI XU, C.J., (Member, IEEE), JIAN WANG, (Member, IEEE), and M. JIAN YUAN, IEEE), AND YONG REN, (Member, IEEE), Information Security on Big Data: Privacy and Data Mining. IEEE Access, 2014.
[5]. Jiawei Han, M.K., Data Mining: Concepts and Techniques. 2nd ed. Data Management Systems, ed. M.R. Jim Gray. 2006, San Francisco,CA,USA: Diane Cerra. 772.
[6]. Huang, C.-M., T.-P. Hong, and S.-J. Horng, Mining knowledge from object-oriented instances. Expert Systems with Applications, 2007. **33**(2): p. 441-450.
[7]. Data Mining concept. [cited 2015 April]; Available from: http://en.wikipedia.org/wiki/Data_mining.
[8]. Jiang, F. and C.K.-S. Leung, A Business Intelligence Solution for Frequent Pattern Mining on Social Networks, in Data Mining Workshop (ICDMW), 2014 IEEE International Conference on. 2014, IEEE. p. 789-796.
[9]. Yan, J.H.H.C.D.X.X., Frequent pattern mining: current status and future directions. Data Min Knowl Disc, 2007: p. 55-77.
[10]. Warnars, S., Mining Frequent Pattern with Attribute Oriented Induction High Level Emerging Pattern (AOI-HEP), in 2nd International Conference on Information and Communication Technology (ICoICT). 2014, IEEE: Tangerang, Indonesia. p. 149-154.
[11]. Sun, S. and J. Zambreno, Design and analysis of a reconfigurable platform for frequent pattern mining. Parallel and Distributed Systems, IEEE Transactions on, 2011. **22**(9): p. 1497-1505.
[12]. Sunil Joshi, D.R.S.J., and Dr. R. C. Jain, An Implementation of Frequent Pattern Mining Algorithm using Dynamic Function. International Journal of Computer Applications, Nov. 2010. **Volume. 9**.
[13]. Jiawei Han, Micheline Kamber, and J. Pei, DATA MINING: Concepts and Techniques. 3rd ed, ed. C. Faloutsos. 2011: China Machine Press. 673 pages.
[14]. Goethals, B., Survey on Frequent Pattern Mining. Finland.
[15]. Nguyen, T.-T. An improved algorithm for frequent patterns mining problem. in Computer Communication Control and Automation (3CA), 2010 International Symposium on. 2010. IEEE.
[16]. Ullman, The Relational data model. Stanford. p. 403-450.
[17]. Codd, E.F., "A relational model of data for large shared data banks". ACM, 1970. **Vol.13 No.6**: p. pp.377-387.
[18]. Codd, E.F., "Derivability, redundancy, and consistency of relations stored in large data banks". IBM Research Report, San Jose., 1969. **Vol.RJ599**.
[19]. Vassilakopoulos, N.N.K.a.M., "Comparison of Post-Relational and Object-Relational modelling for real-World database applications". Journal of Systems and Information Technology, 2014. **Vol.16**(Iss 4): p. pp.313-340.
[20]. Widow, J.D.U.a.J., ed. A First Course in DATABASE SYSTEMS. 3rd ed. 2008, Pearson Education, Inc.
[21]. "Relational Database Model and Object Oriented Model". System Analysis and Design [cited 2015 March]; Available from: http://www.freetutes.com/systemanalysis/sa8-relational-database-model.html.
[22]. S.M Fakhrahmad, G.D., An Efficient Frequent Pattern Mining Method and its Parallelization in Transactional Databases. Journal of Information Science and Engineering, 2011. **Vol.27**: p. pp.511-525.
[23]. Mishra, S., D. Mishra, and S.K. Satapathy, Fuzzy pattern tree approach for mining frequent patterns from gene expression data, in Electronics Computer Technology (ICECT), 2011 3rd International Conference on. 2011, IEEE. p. 359-363.
[24]. Leung, C.K.S., "Uncertain frequent pattern mining" in Frequent pattern mining. October, 2014: p. 417-453.
[25]. C.K.S. leung, C.L.C., P.Johnstone, D.S H.C. yUEN, Interactive visual analytics of databases and frequent sets. International Journal of Information Retrieval Research, 2013. **3**(4): p. 120-140.
[26]. R.Srikant, R.A.a., Fast algorithms for mining association rules Int.Conf.Very Large Data Bases, 1994: p. pp.487-499.
[27]. Zhang, K., et al., A Method to Optimize Apriori Algorithm for Frequent Items Mining, in Computational Intelligence and Design (ISCID), 2014 Seventh International Symposium on. 2014, IEEE. p. 71-75.
[28]. Nasreen, S., et al., Frequent Pattern Mining Algorithms for Finding Associated Frequent Patterns for Data Streams: A Survey. Procedia Computer Science, 2014. **37**(0): p. 109-116.
[29]. Charu C. Aggarwal, J.H., Frequent Pattern Mining. 2014: Switzerland.
[30]. Singh, A., A. Kumar, and A.K. Maurya, An empirical analysis and comparison of apriori and FP-growth algorithm for frequent pattern mining, in Advanced Communication Control and Computing Technologies (ICACCCT), 2014 International Conference on. 2014, IEEE. p. 1599-1602.
[31]. S., X.V.K.J.R.Q.J.G.Q.Y.H.M.G.J.M.A.N.B.L.P., Top 10 algorithms in data mining. Knowl Inf Syst, 2008. **Volume. 14**: p. 1-37.
[32]. J. Han, J.P., Y. Yin, R. Mao, Mining frequent patterns without candidate generation: A frequent-pattern tree approach. Data Mining and Knowledge Discovery, 2004. **8**: p. 53-87.
[33]. Wang, P., C. An, and L. Wang, An improved algorithm for Mining Association Rule in relational database, in Machine Learning and Cybernetics (ICMLC), 2014 International Conference on. 2014, IEEE. p. 247-252.
[34]. Jain, J.K., N. Tiwari, and M. Ramaiya, Mining Positive and Negative Association Rules from Frequent and Infrequent Pattern Using Improved Genetic Algorithm, in Computational Intelligence and Communication Networks (CICN), 2013 5th International Conference on. 2013, IEEE. p. 516-521.
[35]. WeeKeong, Y., Amitabha Das, Rapid Association Rule Mining, in Information and Knowledge Management. 2001: Atlanta, Georgia. p. 474-481.
[36]. Brijendra Dhar Dubey, Mayank Sharma, and R. Shah, COMPARATIVE STUDY OF FREQUENT ITEMSET IN DATA MINING. International Journal of Programming Languages and Applications (IJPLA), 2015. **Vol.5 No.1**.
[37]. Borgelt, C., Finding Frequent Item Sets by Recursive Elimination. 2005.