International Journal for Research in Engineering Application & Management (IJREAM)
ISSN : 2454-9150 Vol-03, Issue-04, July 2017

Improved Algorithm for Mining of High Utility patterns in one phase Based on Map reduce Framework on Hadoop

[1]Ms. Geeta Raju Popalghat, [2]Prof. S. B. Kothari

[1]Student, [2]Professor, [1,2]G.H Raisoni college of engineering and management, Ahmednagar, Maharashtra, India.

[1]geetarpopalghat@gmail.com, [2]s.kothari@raisoni.net

Abstract— Mining high utility itemsets from a value-based database alludes to the disclosure of itemsets with high utility like benefits. In spite of the fact that various significant calculations have been proposed lately, they bring about the problem of causing a sizably voluminous number of applicant itemsets for high utility itemsets. Such a large number of candidate itemsets degrades the mining performance in terms of execution time and space requirement. Earlier work shows this on two phase candidate generation. This approach suffers from scalability issue due to the huge number of candidates. Our paper presents the efficient approach where we can generate high utility patterns in one phase without generating candidates. Here we have taken experiments on linear data structure, our pattern growth approach is to search a reverse set enumeration tree and to prune search space by utility upper bounding. Also high utility patterns are identified by a closure property and singleton property. Iin this venture we are displaying new approach which is extending these calculations to conquer the restrictions utilizing the Map Reduce structure on Hadoop. Experimental results show that the proposed algorithms, not only reduce the number of candidates effectively but also outperform other algorithms substantially in terms of runtime, especially when databases contain lots of long transactions.

Keywords— Data mining, utility mining, high utility patterns, frequent patterns, pattern mining, Map Reduce, Hadoop.

I. INTRODUCTION

Mining frequent itemset is most important problem in data mining. The most commonly used method to find frequent itemset are Apriori and FP-Growth. Apriori algorithm which generates candidate set for mining frequent patterns and FP-Growth which mine frequent patterns without candidate generation. A Processing changeable data streams in real time is one of the most important issues in the data mining field due to its broad applications such as retail market analysis, wireless sensor networks, and stock market prediction. In addition, it is an interesting and challenging problem to deal with the stream data since not only the data have unbounded, continuous, and high speed characteristics but also their environments have limited resources. High utility mining, in the mean time, is one of the basic research themes in design mining to beat real downsides of the conventional structure for frequent pattern mining that takes just double databases and indistinguishable thing significance into thought. This approach conducts mining processes by reflecting characteristics of real world databases, non-binary quantities and relative importance of items. Discovering helpful patterns hidden in a huge database plays an important role in many data processing tasks, few examples are frequent pattern mining, weighted frequent pattern mining, and high utility pattern mining. Among these, frequent pattern mining could be an elementary analysis topic these has been applied to completely different forms of databases, like transactional databases, streaming databases and statistic databases.

Utility mining [4] emerged recently to handle the limitation of frequent pattern mining by considering the user's expectation or goal similarly because the information. Utility mining with the itemset share framework [9], [3], [4], for example, discovering combos of product with high profits or revenues, is far tougher than different classes of utility mining issues, for instance, weighted itemset mining [10], [2], [3] and objective-oriented utility-based association mining [11], [3]. Concretely, the powerfulness measures within the latter classes observe associate anti-monotonicity property, that is, a superset of associate uninteresting pattern is also uninteresting. Such a property will be utilized in pruning search area, that is additionally the use of all frequent pattern mining algorithms [3]. sadly, the anti-monotonicity property doesn't apply to utility mining with the itemset share framework [39], [40]. Therefore, utility mining with the itemset share framework is tougher than the opposite classes of utility mining similarly as frequent pattern mining.

High utility itemset is itemset which have utility no less than a user-specified minimum utility threshold; otherwise, it is called a low-utility itemset. In many applications like cross marketing

in retail stores mining such high utility itemsets from databases is an important task.

Existing techniques [21, 22, 23, 24, 25, 26] used for utility pattern mining. However, the existing methods often generate a large set of potential high utility itemsets and the mining performance is degraded consequently. On the off chance that database contain long exchanges or low edge esteem is set circumstance is more confounded for utility mining. The large number of potential high utility itemsets forms a challenging problem to the mining performance. The challenge is that the amount of candidates will be huge, that is that the measurability and potency bottleneck. Although a great deal of effort has been created [4], [15], [2], [38] to reduce the amount of candidates generated within the 1st phase, the challenge still persists once the information contains several long transactions or the minimum utility threshold is little. Such an enormous range of candidates causes measurability issue not solely within the 1st section however additionally in the second section, and consequently degrades the potency.

To address the challenge, this paper proposes a new algorithm, d2HUP with Hadoop, for utility mining with the itemset share framework, which employs several techniques proposed for mining frequent patterns with Hadoop as parallel processing which will give less time complexity to process such large datasets.

II. REVIEW OF LITERATURE

R. Agarwal, C. Aggarwal, and V. Prasad introduce a calculation for mining long examples in databases [1]. The algorithm discovers expansive itemsets by utilizing depth first search on a lexicographic tree of itemsets.The focus of this paper is to develop CPU-efficient algorithms for finding the frequent itemsets in the cases when the database contains patterns which are very wide. Author refers to this algorithm as Depth Project, and it achieves more than one order of magnitude speedup over the recently proposed MaxMiner algorithm for finding long patterns. These techniques may be quite subsidiary for applications in areas such as computational biology in which the number of records is relatively minuscule, but the itemsets are very long. This necessitates the revelation of patterns utilizing algorithms which are especially tailored to the nature of such domains.

As of late, high utility patterns (HUP) mining is a standout a

mongst the most imperative research issues in information mining because of its competency to consider the nonbinary recurrence estimations of things in transaction and diverse benefit esteems for each item.On the other hand, incremental and interactive data mining provide the faculty to utilize anterior data structures and mining results in order to reduce dispensable calculations when a database is updated, or when the minimum threshold is transmuted. In this paper [4], we propose three novel tree structures to efficiently perform incremental and interactive HUP mining. The main tree structure, Incremental HUP Lexicographic Tree ({\rm IHUP}_{{\rm {L}}}-Tree), is organized by a thing's lexicographic request. It can catch the incremental information with no rebuilding operation. The second tree structure is the IHUP Transaction Frequency Tree ({\rm IHUP}_{{\rm {TF}}}-Tree), which acquires a reduced size by orchestrating things as indicated by their transactions frequency (plunging request To diminish the mining time, the third tree, IHUP-Transaction-Weighted Utilization Tree ({\rm IHUP}_{{\rm {TWU}}}-Tree) is composed predicated on the TWU estimation of things in sliding request. Large performance analyses show that our tree structures are very efficient and scalable for incremental and interactive HUP mining. F. Bonchi, F. Giannotti, A. Mazzanti, and D. Pedreschi presents approach named ExAnte [6] is a simple yet effective for preprocessing input data for mining frequent patterns. The approach questions established research in that it requires no trade-off between antimonotonicity and monotonicity. Indeed, ExAnte relies on a vigorous synergy between these two antithesis components and exploits it to dramatically reduce the data being analyzed to those containing fascinating patterns. This data reduction, in turn, induces a strong reduction of the candidate patterns' search space. The result is paramount performance amendments in subsequent mining. It can withal make feasible some otherwise intractable mining tasks. The authors describe their technology and experiments that proved its effectiveness using different constraints on various data sets.

In the last years, in the context of the constraint-predicated pattern revelation paradigm, properties of constraints have been studied comprehensively and on the substratum of these properties, competent constraint-pushing techniques have been defined. In this paper we review and elongate the state-of-the-art of the constraints that can be pushed in a frequent pattern computation. This paper [8] introduces novel data reduction techniques which are able to exploit convertible anti-monotone constraints (e.g., constraints on average or median) as well as tougher constraints (e.g., constraints on difference or standard deviation). A thorough experimental study is performed and it corroborates that our framework outperforms antecedent algorithms for convertible constraints, and exploit the tougher ones with the same effectiveness. At last, creator highlight that the primary preferred standpoint of this approach, i.e., driving imperatives by methods for information decrease in a level-wise structure, is that distinctive properties of various limitations can be exploited all together, and the aggregate advantage is constantly more prominent than the total of the individual advantages. This exercise leads to the definition of a general Apriori-like algorithm which is able to exploit all possible kinds of constraints studied so far.

High-utility itemset mining is an emerging research area in the field of Data Mining. Many algorithms were proposed to find high-utility itemsets from transaction databases and use a data structure called UP-tree for their working. However, algorithms predicated on UP-tree engender a plethora of candidates due to circumscribed information availability in UP-tree for computing utility value estimates of itemsets. In this paper [12], author present a data structure named UP-Hist tree which maintains a histogram of item quantities with each node of the tree. The histogram allows computation of better utility estimates for efficacious pruning of the search space. Extensive experiments on authentic as well as synthetic datasets show that their algorithm based on UP-Hist tree outperforms the state of the art pattern-magnification predicated algorithms in terms of the add up to number of applicant high utility itemsets created that should be confirmed.

III. PROPOSED SYSTEM

To provide the efficient solution to mine the astronomically immense transactional datasets, recently ameliorated methods presented propose two novel algorithms as well as a compact data structure for efficiently discovering high expediency itemsets from transactional databases. Experimental results show that d2HUP and CAUL outperform other algorithms substantially in terms of execution time. But these algorithms further need to be extend so that system with less memory will also able to handle large datasets efficiently.

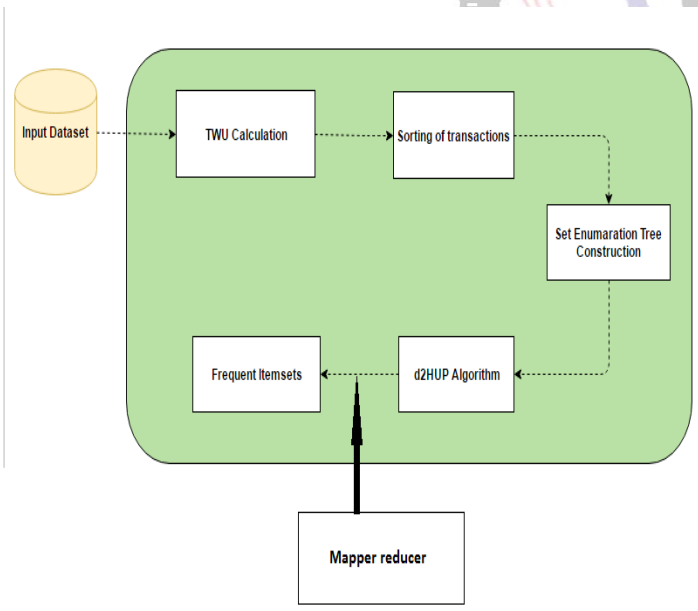

Fig. 1. Block Diagram of Proposed System

As we can see in Fig. 1 it shows the proposed architecture of our system. We will take any transaction dataset of supermarket as a input to find our high utility patterns. First process is preprocessing. After that we will find out TWU of each item of dataset. Based on minimum support value we will do sorting of transaction items whose TWU is greater than mentioned support. Further set enumeration tree will be created so that we can apply d2HUP algorithm on these transactions to find out frequent itemsets. We are proposing here big data hadoop for large transactions. By using this technique we can find out frequent patterns without candidate generation.

The algorithms presented in paper are practically implemented with memory 3.5 GB, but if memory size is 2 GB or below. the performance will again declass in case of time. In this venture we are exhibiting new approach which is extending these calculations to conquer the impediments utilizing the Map Reduce structure on Hadoop.

Algorithm d2HUP

1 build $TS(\{\})$ and Ω from D and XUT
2 $N \leftarrow$ root of reverse set enumeration tree
3 DFS$(N, TS(pat(N)), minU, \Omega)$

Subroutine: DFS$(N, TS(pat(N)), minU, \Omega)$
4 **if** $u(pat(N)) \geq minU$ **then output** $pat(N)$
5 $W \leftarrow \{i | i \prec pat(N) \wedge uB_{item}(i, pat(N)) \geq minU\}$
6 **if** $Closure(pat(N), W, minU)$ is satisfied
7 **then output** nonempty subsets of $W \cup pat(N)$
8 **else if** $Singleton(pat(N), W, minU)$ is satisfied
9 **then output** $W \cup pat(N)$ as an HUP
10 **else foreach** item $i \in W$ in Ω **do**
11 **if** $uB_{fpe}(\{i\} \cup pat(N)) \geq minU$
12 **then** $C \leftarrow$ the child node of N for i
13 $TS(pat(C)) \leftarrow$ Project$(TS(pat(N)), i)$
14 DFS$(C, TS(pat(C)), minU, \Omega)$
15 **end foreach**

d2HUP builds TS({})by scanning the database D and the external utility table XUT to compute s({i}),u({i}), $uB_{item}(i, \{\ \})$, *and* $uB_{fpe}(\{i\})$ for each item i. and makes Ω in the descending order of uB_{item} (at line 1). d2HUP starts searching high utility patterns from the root of reverse set enumeration tree (at lines 2-3) by calling the DFS(*N,TS(pat(N)), minU, Ω*) subroutine.

For the node N currently being visited, DFS prints *pat(N)* as a high utility pattern if its utility is no less than the threshold, makes the set W of relevant items, and then gets through one of the three branches as follows. If the closure property holds, DFS outputs every prefix extension of *pat(N)* with relevant items as a high utility pattern by Theorem 4. If the singleton property holds, DFS prints the union of all the relevant items and *pat(N)* as a high utility pattern by Theorem 5. For each relevant item i ϵ W, if the upper bound on the utilities of prefix extensions of $\{i\} \cup pat(N)$ is no less the threshold, DFS

prepares TS($pat(C)$) for the child node C with *item*(*C*)← *i* and *pat(C)* ← $\{i\} \cup pat(N)$, and recursively searches the subtree rooted at C.

Algorithm CAUL

```
1   foreach relevant item j ≺ i do
2     (s[j], u[j], uB_item[j], uB_fpe[j], link[j]) ← 0
3   end foreach
4   foreach utility list t threaded by link[i] do
5     u(pat(N), t) ← u(pat(P), t) + u(i, t)
6     Σ ← u(pat(N), t)
7     foreach relevant item j ∈ t ∧ j ≺ i by Ω do
8       s[j] ← s[j] + 1
9       u[j] ← u[j] + u(j, t) + u(pat(N), t)
10      Σ ← Σ + u(j, t)
11      uB_fpe[j] ← uB_fpe[j] + Σ
12    end foreach
13    foreach relevant item j ∈ t ∧ j ≺ i by Ω do
14      uB_item[j] ← uB_item[j] + Σ
15      thread t into the chain by link[j]
16    end foreach
17  end foreach
```

In algorithm CAUL For any node N and its parent node P with *pat(N)* = $\{i\} \cup pat(P)$ on the reverse set enumeration tree, TS_{CAUL} (*pat(N)*) can be efficiently computed by a pseudo projection, where the pseudo TS_{CAUL}(*pat(N)*) the same memory space with TS_{CAUL}(*pat(P)*).

The utility lists of the pseudo TS_{CAUL}(*pat(N)*) are delimited by following *link*[*i*] in TS_{CAUL}(*pat(P)*) , and the summary entry for each item j ≺ *i* of the pseudo TS_{CAUL}(*pat(N)*) is computed by scanning each delimited utility list.

IV. RESULT ANALYSIS

We used 6 datasets as T10I6D1M, T20I6D1M, Chess, Chain-store, Foodmart, and WebView-1 respectively.

Table 1 shows the running time by the five algorithms. For example, for T10I6D1M with minU ¼ 0:01%, d2HUP takes 27 seconds, HUIMiner 154, UPþUPG 101, IHUPþTWU 109, and Two-Phase runs out of memory. The observations are as follows.

To start with, d2HUP is up to 1 to 3 requests of extent more productive than UPþUPG, IHUPþTWU, and TwoPhase. Specifically, d2HUP is up to 6.6, 6.8, 7.8, 261, 472, and 1,502 times speedier than UPþUPG on T20I6D1M, T10I6D1M, Chain-store, Web-View-1, Foodmart, and Chess individually.

TABLE I Running Time for All Algorithm with minU 0.01%

Algorithm / Dataset	d2HUP (sec)	HUIM. (sec)	UP+ (sec)	IHUP+ (sec)	T.P. (sec)	Proposed Algorithm (sec)
T20I6D1M	27	154	101	109	OM	20
T10I6D1M	35	159	122	111	OM	31
Chain-store	21	151	98	101	OM	20
Web-View-1	67	201	154	167	OM	66
Foodmart	98	222	178	156	OM	67
Chess	56	167	156	145	OM	51

V. CONCLUSION

In this paper we have worked on two algorithms, d2HUP, for utility mining, which finds high utility patterns without generating candidates. We have used linear data structure, CAUL, which mainly focuses on two phase candidate generation approach which was earlier used by prior algorithms. Here pattern enumeration strategy is integrated by high utility pattern growth approach and CAUL. It shows that the strategies considerably improved performance by reducing both the search space and the number of candidates. In this project we are presenting new approach which is extending these calculations to defeat the restrictions utilizing the MapReduce system on Hadoop.

In future we will work on more efficient and enhanced algorithms on different datasets.

REFERENCES

[1] R. Agarwal, C. Aggarwal, and V. Prasad, "Depth first generation of long patterns," in Proc. ACM SIGKDD Int. Conf. Knowl. DiscoveryData Mining, 2000, pp. 108–118.

[2] R. Agrawal, T. Imielinski, and A. Swami, "*Mining association rules between sets of items in large databases*," in Proc. ACM SIGMOD Int. Conf. Manage. Data, 1993, pp. 207–216.

[3] R. Agrawal and R. Srikant, "*Fast algorithms for mining association rules*," in Proc. 20th Int. Conf. Very Large Databases, 1994, pp. 487–499.

[4] C. F. Ahmed, S. K. Tanbeer, B.-S. Jeong, and Y.-K. Lee, "*Efficient tree structures for high utility pattern mining in incremental databases*," IEEE Trans. Knowl. Data Eng., vol. 21, no. 12, pp. 1708– 1721, Dec. 2009.

[5] R. Bayardo and R. Agrawal, "*Mining the most interesting rules*," in Proc. 5th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, 1999, pp. 145–154.

[6] F. Bonchi, F. Giannotti, A. Mazzanti, and D. Pedreschi, "*ExAnte: A preprocessing method for frequent-pattern mining*," IEEE Intell. Syst., vol. 20, no. 3, pp. 25–31, May/Jun. 2005.

[7] F. Bonchi and B. Goethals, "*FP-Bonsai: The art of growing and pruning small FP-trees*," in Proc. 8th Pacific-Asia Conf. Adv. Knowl. Discovery Data Mining, 2004, pp. 155–160.

[8] F. Bonchi and C. Lucchese, "*Extending the state-of-the-art of constraint-based pattern discovery*," Data Knowl. Eng., vol. 60, no. 2, pp. 377–399, 2007.

[9] C. Bucila, J. Gehrke, D. Kifer, and W. M. White, "*Dualminer: A dual-pruning algorithm for itemsets with constraints*," Data Mining Knowl. Discovery, vol. 7, no. 3, pp. 241–272, 2003.

[10] C. H. Cai, A. W. C. Fu, C. H. Cheng, and W. W. Kwong, "*Mining association rules with weighted items*," in Proc. Int. Database Eng. Appl. Symp., 1998, pp. 68–77.

[11] R. Chan, Q. Yang, and Y. Shen, "*Mining high utility itemsets*," in Proc. Int. Conf. Data Mining, 2003, pp. 19–26.

[12] S. Dawar and V. Goyal, "*UP-Hist tree: An efficient data structure for mining high utility patterns from transaction databases*," in Proc. 19th Int. Database Eng. Appl. Symp., 2015, pp. 56–61.

[13] T. De Bie, "*Maximum entropy models and subjective interestingness: An application to tiles in binary databases*," Data Mining Knowl. Discovery, vol. 23, no. 3, pp. 407–446, 2011.

[14] L. De Raedt, T. Guns, and S. Nijssen, "*Constraint programming for itemset mining*," in Proc. ACM SIGKDD, 2008, pp. 204–212.

[15] A. Erwin, R. P. Gopalan, and N. R. Achuthan, "*Efficient mining of high utility itemsets from large datasets*," in Proc. 12th Pacific-Asia Conf. Adv. Knowl. Discovery Data Mining, 2008, pp. 554–561.

[16] P. Fournier-Viger, C.-W. Wu, S. Zida, and V. S. Tseng, "*FHM: Faster high-utility itemset mining using estimated utility cooccurrence pruning*," in Proc. 21st Int. Symp. Found. Intell. Syst., 2014, pp. 83–92.

[17] L. Geng and H. J. Hamilton, "*Interestingness measures for data mining: A survey*," ACM Comput. Surveys, vol. 38, no. 3, p. 9, 2006.

[18] J. Han, J. Pei, and Y. Yin, "*Mining frequent patterns without candidate generation*," in Proc. ACM SIGMOD Int. Conf. Manage. Data, 2000, pp. 1–12.

[19] R. J. Hilderman, C. L. Carter, H. J. Hamilton, and N. Cercone, "*Mining market basket data using share measures and characterized itemsets*," in Proc. PAKDD, 1998, pp. 72–86.

[20] R. J. Hilderman and H. J. Hamilton, "*Measuring the interestingness of discovered knowledge: A principled approach*," Intell. Data Anal., vol. 7, no. 4, pp. 347–382, 2003.

[21] Chowdhury Farhan Ahmed, Syed Khairuzzaman Tanbeer, Byeong-Soo Jeong, and Young-Koo Lee, Member, IEEE "*Efficient Tree Structures for High Utility Pattern Mining inIncremental Databases*" IEEE Trans. Knowledge and Data Engineering, Vol. 21, Issue 12, pp. 1708-1721, December 2009.

[22] Alva Erwin, Raj P. Gopalan, and N. R. Achuthan, "Efficient Mining of High Utility Itemsets from Large Datasets", In Proc. of PAKDD 2008.

[23] Shankar, S.; Purusothaman, T.; Jayanthi, S. "*Novel algorithm for mining high utility itemsets*" International Conference on Computing, Communication and Networking, Dec. 2008.

[24] Raymond Chan; Qiang Yang; Yi-Dong Shen, "*Mining high utility itemsets*" In Proc. of Third IEEE Int'l Conf. on Data Mining ,November 2003.

[25] Ramaraju, C., Savarimuthu N. "*A conditional tree based novel algorithm for high utility itemset mining*", International Conference on Data mining, June 2011.

[26] Ying Liu, Wei-keng Liao, Alok Choudhary "*A Fast High Utility Itemsets Mining Algorithm*" In Proc. of the Utility-Based Data Mining Workshop, 2005.

