ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 4, Issue 12, December 2016

Mining High Utility Patterns in One Phase without Generating Candidate Using Big Data

Shivani Tathe[1], Prof.Sonali Patil[2]

PG Student, Dept. of Computer, Bhivarabai Sawant Institute of Technology and Research, India[1]

Assistant Professor, Bhivarabai Sawant Institute of Technology and Research India, Pune India[2]

ABSTRACT: Utility mining is a new development of data mining technology. Prior works on this problem all employ a two-phase, candidate generation approach with one exception that is however inefficient and not scalable with large databases. The two-phase approach lacks due to the scalability issue due to the huge number of candidates. This paper proposes a novel algorithm that finds high utility patterns in a single phase without generating candidates using Big Data Technology. Firstly, this pattern growth approach is to search a reverse set enumeration tree. Also, have the precision to look ahead to identify high utility patterns without enumeration by a closure property and a singleton property. This linear data structure enables to compute a tight bound for powerful unpleasant and to directly identify high utility patterns by an efficient and scalable way, which targets the root cause with prior algorithms and give an effective solution to the 2-phase candidate generation problem.

KEYWORDS:Utility mining , pattern mining, big data.

I. INTRODUCTION

Finding interesting patterns has been an important data mining task, and has a variety of applications like inventory prediction, where interestingness measures play an important role. With frequent pattern mining a pattern is regarded as interesting if its occurrence frequency exceeds a user specified threshold. For example, mining frequent patterns from a shopping transaction database leads to the discovery of sets of products that are frequently purchased together by customers. However, a user's interest may relate to many factors that are not always expressed in terms of the occurrence frequency. For example, a supermarket manager may be interested in discovering combinations of products with high profits or revenues, which talks about the unit profits and purchased quantities of products that are not considered in frequent pattern mining.Utility mining emerged recently to address the limitation of frequent pattern mining by considering the user's expectation or goal as well as the raw data. Utility mining with the itemset share framework for example, discovering combinations of products with high profits/revenues, is hard than other categories of utility mining problems, for example, weighted itemset mining and objective-oriented utility-based association mining. Concretely, the interestingness measures in the latter categories observe an anti-monotonicity property, that is, a superset of an uninteresting pattern is also uninteresting. Such a property can be employed in reducing search space, which is also the foundation of all frequent pattern mining algorithms. Unfortunately, the anti-monotonicity property does not apply to utility mining with the itemset share framework. Therefore, utility mining with the itemset share framework is more challenging than the other categories of utility mining as well as frequent pattern mining.

Majority of the prior utility mining algorithms with the itemset share framework adopt a two-phase, candidate generation approach, that is, first find candidates of high utility patterns in the first phase, and scan the raw data one more time to identify high utility patterns from the candidates in the second phase.The number of candidates can be huge which poses a challenge, which is the scalability and efficiency bottleneck. Although a lot of effort has been made to reduce the number of candidates generated in the first phase, the challenge persists when the raw data contains numerous long transactions or the minimum utility threshold is small. Such a huge number of candidates causes scalability issue not only in the first phase but also in the second phase, and consequently degrades the efficiency. An exception is the HUIMiner algorithm which is however even less efficient than two-phase algorithms when mining large databases due to inefficient join operations, lack of strong reduction, and scalability issue with its vertical data structure.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 4, Issue 12, December 2016

II. SIMULATION RESULT

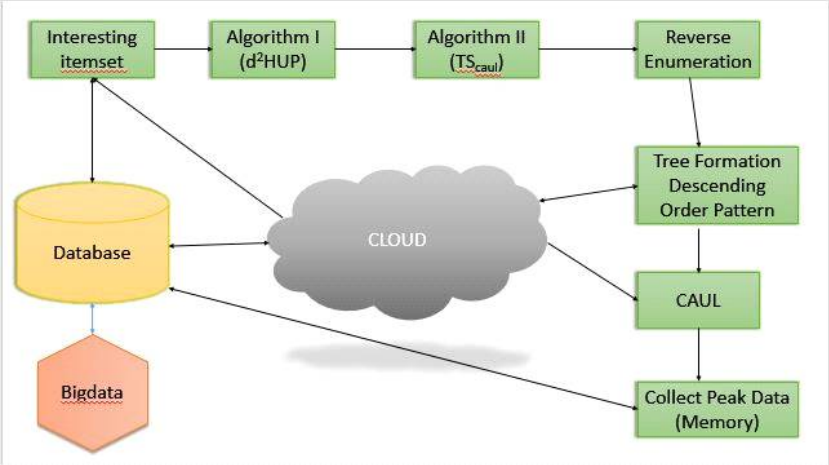

Fig. 1. Block Diagram

The paper (Mining High Utility Pattern in One Phase without Generating Candidates. Junqiang Liu, Member, IEEE, Ke Wang, Senior Member, IEEE, and Benjamin C.M. Fung, Senior Member, IEEE) proposes a new algorithm, d2HUP, for utility mining with the itemset share framework which finds high utility patterns without candidate generation. It contributions include: 1) A linear data structure, CAUL, is proposed, which targets the root cause of the two phase, candidate generation approach used by prior algorithms, that is, their data structures cannot keep the original utility information. 2) A high utility pattern growth approach, which integrates a pattern enumeration strategy, pruning by utility upper bounding, and CAUL. This basic approach outperforms prior algorithms strikingly. 3) Our approach is enhanced significantly by the look ahead strategy that identifies high utility patterns without enumeration. In the future, we will work on high utility sequential pattern mining, parallel and distributed algorithms, and their application in big data analytics.

III. SYSTEM ARCHITECTURE / SYSTEM OVERVIEW

1. Big Data: Big data is a term that describes a large volume of data both structured and unstructured that inundates a business on a day-to-day basis. But it's not the amount of data that's important. Its what organizations do with the data that matters. Big data can be analyzed for insights that lead to better decisions and strategic business moves.

2. Reverse Enumeration: Enumeration Tree gets generated in this phase manipulating data.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 4, Issue 12, December 2016

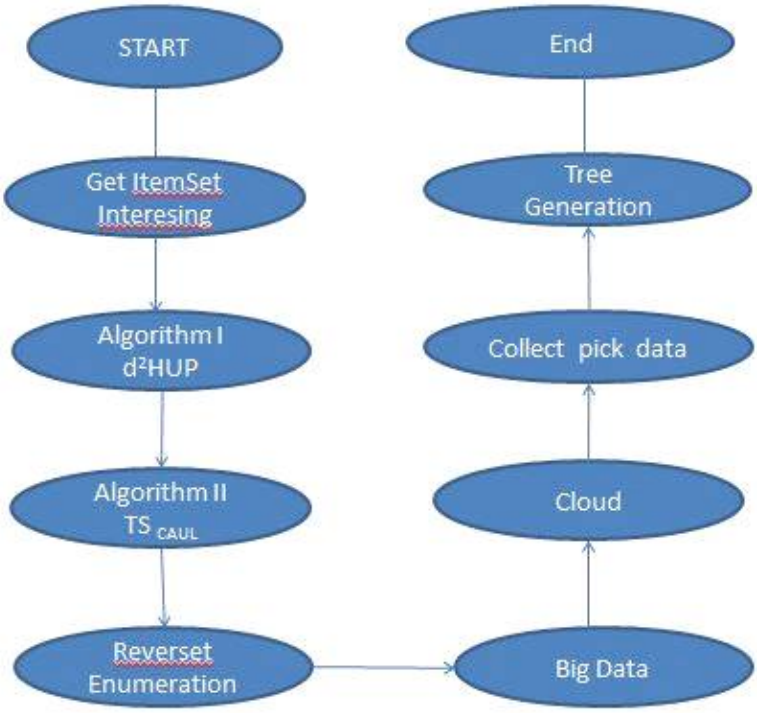

FIG. 2. FLOW CHART

3. Tree formation in Descending order pattern: The tree is formed per the data retrieved from the reverse set Enumeration which get the data from the Algorithms implemented. The data is in descending order hence it is easy to separate the data from the unfrequently occurred data.

4. CAUL (Chain of Accurate Utility List): The Chain of accurate utility list is given to the tree for the accurate result of the data set enumeration. In this case the CAUL is required to categorized the frequent data from infrequent data.

5. Collect Pick Data Memory: The data memory is saved here and giving to the data base at this stage.

IV. ALGORITHM PROPOSED

1. Algo1: d^2HUP 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 by Definitions 2 and 3, Corollary 2, and Theorem 1 or Corollary 3, and makes Ω in the descending order of uB_{item} (at line 1). d^2HUP starts searching high utility patterns from the root of reverse set enumeration tree (at lines 2-3) by 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 (at line 4), makes the set W of relevant items (at line 5), and then gets through one of the three branches as follows. If the closure property holds, DFS outputs every prefix extension pat(N) of with relevant items as a high utility pattern by Theorem 4 (at lines 6-7); If the singleton property,

Algorithm 1: HU P Algorithm
1: Build TS ({}) and Ω from D and X ∪ T
2: N ← root of reverse set enumeration tree

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 4, Issue 12, December 2016

3: DFS (N, TS(pat(N)), minU, Ω)
4: **Subroutine:** DFS (N, TS(pat(N)), minU, Ω)
5: **if**u(pat(N)) ≥minU**then** output pat(N)
6: W ← {i|i≺pat(N) ^ uB_{item}(I, pat(N)) ≥minU}
7: **if**Closure(pat(N), W, minU) is satisfied
8: **then** output nonempty subsets of W ∪pat(N)
9: **else if** singleton (pat(N), W, minU) is satisfied
10: **then** output W ∪pat(N) as an H∪ P
11: **else foreach** item i∈W in Ω**do**
12: if uB_{fpe}({i}∪pat(N)) ≥minU
13: **then** C ← the child node of N for i
14: TS(pat(C)) ← Project(TS(pat(N)), i)
15: DFS (C, TS (pat(C)), minU, Ω)
16: **end foreach**

DFS prints the union of all the relevant items and pat(N) as a high utility pattern by Theorem 5 (at lines 8-9). For each relevant item i∈ W, if the upper bound on the utilities of prefix extensions of {i} ∪ 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} ∩ pat(N) recursively searches the subtree rooted at C. Note that DFS computes TS(pact(C)) by a pseudo projection from TS(pat(N)) which is implemented in Algorithm 2.

Algorithm 2: Pseudo Project
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: ∑←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: ∑←∑ + u (j, t)
11: uB_{fpe}[j] ←uB_{fpse}[j] + ∑
12: **end foreach**
13: **foreach** relevant item j∈ t ^ j≺i by Ω**do**
14: uB_{item}[j] ←uB_{item}[j] + ∑
15: thread t into the chain by link[j]
16: **end foreach**
17: **end foreach**

2. Algo2: For any node N and its parent node P with pat(N) = {i} ∪ 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)) shares 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. This computation process is implemented by Algorithm 2, namely PseudoProject, which is called by d^2HUP at line 13 in Algorithm 1. We can do pseudo projection recursively in two senses. First, TS_{caul}(pat(P)) for a

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 4, Issue 12, December 2016

reverse set enumeration tree node P can be projected to the pseudo TS_{caul}(pat(N)) for a childnode N of P, which in turn can be projected to the pseudo TS_{caul}(pat(C)) for a grandchild node C of P.

V. PSUEDO CODE

Thecentraloutlineoftheproposedalgorithmisasfollows.

- 1:Select K points as the initial centroids (Item IDs)
- 2:REPEAT
- 3:Form K clusters by assigning all points to the closest centroid (done with the help of spark programming)
- 4:Recompute the centroid of each cluster
- 5:UNTIL
- the centroids don't change.
- 6:All users are weighted with respect to similarity with the active user Similarity between users is measured as the Pearson correlation between their ratings vectors. (ranging from 1 to 5.)
- 7:Selectn active users that have the highest similarity.
- 8:Compute a prediction P a , u from a weighted combination .Similarity between two users is computed using the Pearson correlation coefficient. (data values)
- 9:Apply content filtering and collaborative filtering

VI. CONCLUSION

The resent base paper is introducing the 1 phase candidate generation technique which is fast and efficient. This can be done by using big data for more information to be gained and collected. From the vendor perspective, it is a new, on demand, pay-per-use, and highly flexible, delivery and provisioning system for IT related resources and services. Thus, this paper summarizes mining high utility patterns in one phase using big data.

REFERENCES

[1] K. Pahlavan,Feritakgul and Y.YE,Taking Positioning Indoor Wi-Fi Localization and GNSS, Inside GNSS Journal May 2010: 40-47.
[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. 207216
[3] M. J. Zaki and C. Hsiao, Efficient algorithms for mining closed itemsets and their lattice structure, IEEE Trans. Knowl. Data Eng., vol. 17, no. 4, pp. 462478, Apr. 2005.
[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. 17081721, Dec. 2009.
[5] G.-C. Lan, T.-P. Hong, and V. S. Tseng, an efficient projection based indexing approach for mining high utility itemsets, Knowl. Inf. Syst., vol. 38, no. 1, pp. 85107, 2014.
[6] V. S. Tseng, B.-E. Shie, C.-W. Wu, and P. S. Yu, Efficient algorithms for mining high utility itemsets from transactional databases, IEEE Trans. Knowl. Data Eng., vol. 25, no. 8, pp. 17721786, Aug. 2013.
[7] S. Krishnamoorthy, Pruning strategies for mining high utility itemsets, Expert Syst. Appl., vol. 42, no. 5, pp. 23712381, 2015