Efficient Data Mining Utility Patterns without Candidate Generation

Mr. Sanadi Rajesh A[1] Prof. Dhainje P. B[2]
[1]ME Scholar [2]Vice-Principal
[1,2]Department of Computer Science & Engineering
[1,2]Shriram Institute of Engineering & Technology Center, Paniv, Maharashtra, India

Abstract— This paper proposes an algorithm which finds high utility patterns in a single phase without generating candidates. The novelties lie in a high utility pattern growth approach; it's a look ahead strategy, and a linear data structure. Our pattern growth approach is to search a reverse set enumeration tree and to prune search space by utility upper bounding. We look ahead to identify high utility patterns without enumeration by a closure property and a singleton property. Our linear data structure enables us to compute a tight bound for powerful pruning and to directly identify high utility patterns in an efficient and scalable way that targets the root cause with prior algorithms.

Key words: Data Mining, Utility Mining, High Utility Patterns, Frequent Patterns, Pattern Mining, Association, Clustering, Data Mining Application, Knowledge Discovery Database.

I. INTRODUCTION

Finding interesting patterns and searching expected patterns is an important data mining task and also has a variety of applications. E.g. in case of inventory management or shopping analysis where huge dataset is available, it's important for pattern analysis, inventory predictions and particular conditions monitoring. In case of frequent

Pattern mining a pattern is regarded as interesting if its occurrence frequency exceeds a user specified threshold. Searching frequent patterns from a shopping transaction database is to invent sets of products which are frequently purchased together by various customers. e.g. a supermarket manager may be interested to discover different combinations of products with high profits or revenues, which relates to the unit profits and purchased quantities of products that are not considered in frequent pattern mining

To address the challenge, this paper proposes a new Algorithm for utility mining with the item set share framework, that employs several techniques proposed for mining frequent patterns and they are as follows:

- A high utility pattern growth approach is proposed, That argue is one without candidate generation because while the two-phase, candidate generation approach employed by prior algorithms first generates high TWU patterns (candidates) with TWU being an interim, anti-monotone measure and then identifies high utility patterns from high TWU patterns, our approach directly discovers high utility patterns in a single phase without generating high TWU patterns (candidates).
- A look ahead strategy is incorporated with our approach, which tries to identify high utility patterns earlier without recursive enumeration. Such a strategy is based on a closure property and a singleton property, and enhances the efficiency in dealing with dense data.
- A linear data structure, CAUL, is proposed to represent original utility information in raw data, which targets the root cause with prior algorithms, that is, they all employ a data structure to maintain the utility estimates instead of the original utility information, and thus can only determine the candidacy of a pattern but not the actual utility of the pattern in their first phase.

II. RELATED WORKS

High utility pattern mining problem is closely related to frequent Pattern mining, including constraint-based mining. We just briefly review prior works both on frequent pattern mining and on utility mining, and discuss how our work connects to and differs from the prior works.

A. Frequent Pattern Mining

Frequent pattern mining means to discover all patterns whose supports is just like a user-defined minimum support threshold. Frequent pattern mining employs the anti-monotonicity property:

The support of a superset of a pattern is no more than the support of the pattern. Algorithms for both mining frequent patterns and for mining high utility patterns falls into three categories, breadth-first search, depth first search, and hybrid search. Apriori by Agrawal and Srikant is a famous Breadth-first algorithm for mining frequent patterns that scans the disk-resident database as many times as the maximum length of frequent patterns. FP-growth by Han et al. is a well-known depth-first algorithm, which compresses the database by FP-trees in main memory. Eclat by Zaki is a famous hybrid algorithm. It keeps a database or a database partition in memory by a vertical tid-list layout and can work in either depth-first or breadth first manner. This paper adopts a depth-first strategy since breadth first search is typically more memory-intensive and more likely to exhaust main memory and thus slower. Concretely, our algorithm depth-first searches a reverse set enumeration tree, which can be thought of as exploring a regular set enumeration tree right-to-left in a reverse lexicographic order. While Eclat also explores such an order, our algorithm is the first fully exploiting the benefit in mining high utility patterns.

B. Constraint-Based Mining

Constraint-based mining is a milestone in evolving from frequent pattern mining to utility mining. Works on this area mainly focus on how to push constraints into frequent pattern mining algorithms.

Pei et al. discussed constraints that are similar to (normalized) weighted supports, and first observed an interesting property, called convertible anti-monotonicity, by arranging the items in weight-descending order. The authors demonstrated how to push them into the FP-growth algorithm. Bucila et al. considered mining patterns that satisfy a conjunction of anti-monotone and monotone constraints, and proposed an algorithm, DualMiner, that efficiently prunes its search space using both anti-monotone and monotone constraints. Bonchi et al. introduced the

All rights reserved by www.ijsrd.com

ExAnte property which states that any transaction that does not satisfy the given monotone constraint can be removed from the input database, and integrated the property with Apriori-style algorithms. Bonchi and Goethals applied the ExAnte property with the FP-growth algorithm. Bonchi and Lucchese generalized the data reduction technique to a unified framework. De Raedt et al. investigated how standard constraint programming techniques can be applied to constraint-based mining problems with constraints that are monotone, antimonotone, and convertible. Bayardo and Agrawal, and Morishita and Sese proposed techniques of pruning based on upper bounds when the constraint is neither monotone, anti-monotone, nor convertible. This paper also employs such a standard technique. Our contribution is to develop tight upper bounds on the utility.

III. HIGH UTILITY PATTERN GROWTH

We introduce a reverse set enumeration tree as a way to enumerate patterns, and then propose strong pruning techniques that drastically.

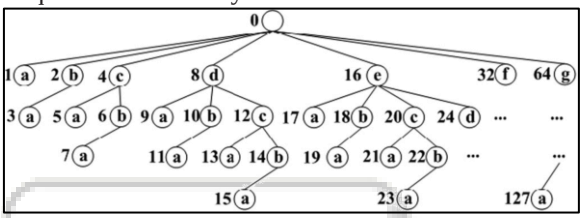

Fig. 1: Growing Reverse Set Enumeration Tree

Our pattern growth approach can be thought of as growing or searching a reverse set enumeration tree in a depth-first manner. The construction of the reverse set enumeration tree follows an imposed ordering V of items. Concretely, the root is labeled by no item, each node N other than the root is labeled by an item, the path from N to the root represents a pattern, and the child nodes of N are labeled by items listed in V. It follows that the sequence of items along the path from N to the root is in accordance with V. The imposed ordering of items, denoted by V, is a pre-determined, ordered sequence of all the items in I.

Accordingly, for items i and j, i _ j denotes that i is listed before j; i _ X denotes that i _ j for every j 2 X, and W _ X denotes that i _ X for every i 2 W, in accordance with V. The imposed ordering V of items can be determined by a heuristic proposed. Given V, a pattern can also be represented as an ordered sequence. For brevity, we use the set notation, for example, {a, b, c}, in place of the sequence notation, for example, <a, b, c>. For example the imposed ordering is the lexicographic order, i.e., V ¼ fa, b, c, d, e, f, gg, then a _ b, a _ c, a _ fb, cg, fa, bg _ fc, dg, and so on.

The reverse set enumeration tree is equivalent to a regular set enumeration tree that is imposed with a reverse lexicographic order and explored right-to-left, which yields a property: a pattern is always enumerated before its supersets in a depth-first search. For example, fag and fbg are before fa, bg, and fa, bg and fcg before fa, b, cg. Most importantly, by such a construction, the transaction set supporting the enumerated pattern can be determined.

A. Pruning by Utility Upper Bounding

It is computationally infeasible to enumerate all patterns, and a standard technique is to prune the search space. However, for utility mining with the itemset share framework, no anti-monotonicity property can be employed for pruning. An alternative is pruning based on utility upper bounding With our pattern growth approach, it is to estimate an upper bound on utilities of all possible patterns represented by nodes in the subtree rooted at the node currently being explored, when growing the reverse set enumeration tree. If such an upper bound is less than minU, the subtree can be pruned as all patterns in the subtree are not high utility patterns.

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

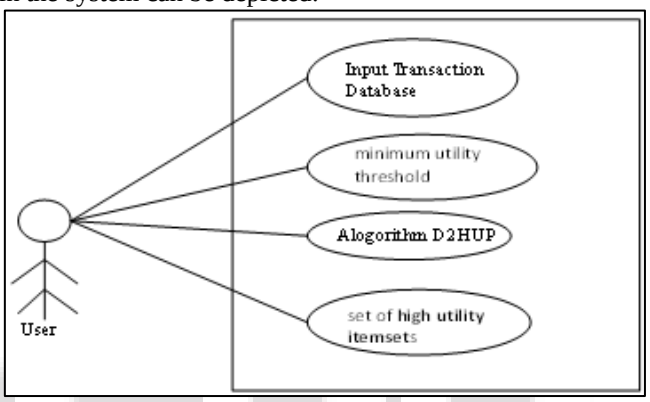

Fig. 2: UML

IV. MINING PATTERNS IN ONE PHASE WITHOUT CANDIDATE GENERATION

The algorithm namely Direct Discovery of High Utility Patterns, which is an integration of the depth-first search of the reverse set enumeration tree, the pruning techniques that drastically reduces the number of patterns to be enumerated, and a novel data structure that enables efficient computation of utilities and upper bounds.

The algorithm lists items in the descending order of uB item based on a heuristic. The pseudo code of algorithm is as follows.

A. Algorithm 1 d2HUP (D, XUT, minU)

1) build TS({}) and V from D and XUT
2) N root of reverse set enumeration tree
3) DFS(N, TS(pat(N)), minU, Ω)
   Subroutine: DFS(N, TS(pat(N)), minU, Ω)
4) if (u(pat)N)) minU then output pat(N)
5) W ← {i|i<pat(N) ^ uBitem(i,pat(N)) minU}
6) if Closure(pat(N), W, minU) is satisfied
7) then output nonempty subsets of W Upat(N)
8) else if Singleton(pat(N), W, minU) is satisfied
9) then output W U pat(N) as an HUP
10) else foreach item i € W in Ω do
11) if $_{u}B_{fpe}$({i} U pat(N))minU
12) then C ← the child node of N for i
13) TS(pat(C)) Project(TS(pat(N)), i)
14) DFS(C, TS(pat(C)), minU, Ω)
15) end foreach

All rights reserved by *www.ijsrd.com*

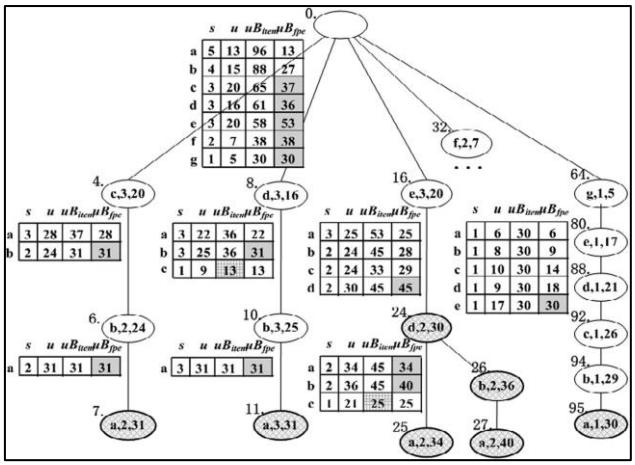

Fig. 3: Extract useful information from large amounts of data

The whole point of the algorithm (and data mining, in general) is to extract useful information from large amounts of data. For example, the information that a customer who purchases a keyboard also tends to buy a mouse at the same time is acquired from the association rule below: Support: The percentage of task-relevant data transactions for which the pattern is true.

Support (Keyboard -> Mouse) =

$$\frac{\text{No. of transacti ons containing both Keyboard and Mouse}}{\text{No. of total transacti ons}}$$

1) Confidence

The measure of certainty or trustworthiness associated with each discovered pattern.

Confidence (Keyboard -> Mouse) =

$$\frac{\text{No. of transacti ons containing both Keyboard and Mouse}}{\text{No. of transacti ons containing (Keyboard)}}$$

The algorithm aims to find the rules which satisfy both a minimum support threshold and a minimum confidence threshold (Strong Rules).

2) Item

Article in the basket.

3) Item set

A group of items purchased together in a single transaction.

V. CONCLUSION

This paper proposes an algorithm for utility mining with the item set share framework, which finds high utility patterns without candidate generation. Our contributions include: A linear data structure is proposed, which targets the root cause of the two phases, candidate generation approach adopted by prior algorithm. Another approach is enhanced significantly by the look ahead strategy that identifies high utility patterns without enumeration.

REFERENCES

[1] Yongjian Fu "data mining: task, techniques and application".
[2] Er. RimmyChuchra "Use of Data Mining Techniques for the Evaluation of Student Performance: A Case Study" International Journal of Computer Science and Management Research Vol 1 Issue 3 October 2012
[3] J. Han and M. Kamber. "Data Mining, Concepts and Techniques", Morgan Kaufmann, 2000.
[4] AakankshaBhatnagar, Shweta P. Jadye, Madan Mohan Nagar" Data Mining Techniques & Distinct Applications: A Literature Review" International Journal of Engineering Research & Technology (IJERT) Vol. 1 Issue 9, November- 2012.
[5] Brijesh Kumar Baradwaj, Saurabh Pal "Mining Educational Data to Analyze Students Performance" (IJACSA) International Journal of Advanced Computer Science and Applications, Vol. 2, No. 6, 2011.
[6] Data mining white paper, www.ikanow.com
[7] Nikita Jain, Vishal Srivastava "Data Mining Techniques: A Survey Paper" IJRET: International Journal of Research in Engineering and Technology, Volume: 02 Issue: 11 | Nov-2013.
[8] Dr. M.H.Dunham, "Data Mining, Introductory and Advanced Topics", Prentice Hall, 2002.
[9] Umamaheswari. K, S. Niraimathi "A Study on Student Data Analysis Using Data Mining Techniques" International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 8, August 2013.
[10] David L Olson, DursunDelen "Advance data minig techniques" springer 2008
[11] G. V. Otari, Dr. R. V. Kulkarni, "A Review of Application of Data Mining in Earthquake Prediction" G. V. Otari et al, / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 3 (2) , 2012,3570-3574
[12] D Ramesh, B Vishnu Vardhan, "Data Mining Techniques and Applications to Agricultural Yield Data" International Journal of Advanced Research in Computer and Communication Engineering Vol. 2, Issue 9, September 2013.
[13] Ruxandra-Ştefania PETRE, "Data mining in Cloud Computing" Database Systems Journal vol. III, no. 3/2012.
[14] BhagyashreeAmbulkar and VaishaliBorkar, "Data Mining in Cloud Computing", MPGI National Multi Conference 2012 (MPGINMC-2012), 7-8 April 2012, Link http://research.ijcaonline.org/ncrtc/number6/mpginmc1047.pdf.

All rights reserved by *www.ijsrd.com*