A Parallel Approach For High Utility Patterns Mining From Distributed Databases

Ms. Ruchi Patel, Assistant Professor, Department of Information Technology
Gyan Ganga Institute of Technology and Sciences, Jabalpur

Abstract

In recent years, the problem of high utility pattern mining become one of the most important research area in data mining. Traditional pattern mining algorithms may not find some most profitable, high priced patterns, due to their lower support. These algorithms reflect only statistical correlation, but it does not reflect semantic significance of the pattern. This gives reason to develop a mining model to find itemsets, which contributes to business organization with high profit. Hence, utility-based pattern mining technique has evolved and got much popularity in recent time. But all of the existing utility pattern mining algorithms are based on centralized database and today's internet era databases are inherently distributed. This inherent distribution source of data and the voluminous in size emerges to develop scalable parallel and distributed algorithm for pattern mining. This paper proposed a parallel and distributed method for mining high utility patterns and also prune the irrelevant data or items. This method is designed in such a way so that it can efficiently generate high utility itemsets with less execution time in distributed environment.

***Keywords*— Frequent pattern mining, High utility pattern mining, Distributed Database, Pruning.**

1. Introduction

Mining frequent itemsets from a transaction database is a fundamental task for knowledge discovery such as association rules, sequential patterns and classification. In the past, numerous methods were proposed to discover frequent itemsets. Among them, the most two famous kinds were level-wise algorithms and pattern-growth methods. These approaches, however, only considered whether an item was bought in a transaction or not. Thus, frequent itemsets just reveal the frequency of occurrence of the itemsets, but do not reflect any other factors, such as price or profit. Thus, frequent pattern mining has following 2 limitations:

1. First it treats all items with the same importance/ weight/price.
2. Second, in one transaction each item appears in a binary (0/1) form, i.e. either present or absent.

In the real world, however, each item in the supermarket has a different importance price and one customer can buy multiple copies of an item. This gives motivation to develop a mining model to discover itemsets, which contributes to business organization with high profit. Recently, a Utility Mining Model (UMM) was defined to solve limitations of frequent pattern mining. This model allows users to express their preference or expectations regarding each item in terms of weight or utility values, and find patterns above the user specified minimum utility threshold.

In some situations, frequent itemsets may only contribute a small portion to the overall profit, while non-frequent ones may contribute a large portion to the profit. For example, sale of diamonds may occur less frequently than that of clothing in department store, but the former gives a much higher profit per unit sold than the latter. Only frequency is thus not sufficient to identify the items which are highly profitable or have other potential effects.

However, high profit items are always purchased rarely. If we just consider the purchased frequencies of itemsets, then high profit itemsets may not be discovered. For example, the profit of television is much higher than milk, but the purchased frequency of television is much less than milk.

Nevertheless, the profits for items should be related to the purchased quantities of the items. If purchased quantity for a low profit item is large, then the total profit for the item will increase. Hence, both profits and purchased quantities for items should be considered.

All of the existing utility-based pattern mining algorithms are considered the centralized database but today's internet era databases are inherently distributed. Most of the organizations operate business in global markets require to perform data mining on distributed data sources to turn them into realistic and meaningful knowledge for their future use and the volume of data available for usage is very high. This inherent distribution source of data and the voluminous in size emerges to develop large-scale parallel and distributed high utility patterns mining.

In this paper, a parallel approach for high utility patterns mining is proposed which generates high priced itemsets from large distributed database. It can also prune irrelevant itemsets which has low utility through downword closure property. In this approach, for distributed environment one master node and some slave nodes are there according to requirement. Very large database is distributed to number of slave nodes. Each node scan its local database and generates the frequent itemsets using A-Priori algorithm then its corresponding gain value is computed. Based on this gain value, the high utility itemsets are mined according to the user specified threshold send it to master node. It also prunes the items that do not satisfied the given threshold. Finally, global high utility pattern are cached by the Master node.

The rest of the paper is organized as follows. Section 2 presents the review of some related research works. Section 3 describes terms and definitions and Section 4 presents the proposed framework and algorithm in details. Finally Section 5 concludes the paper.

2. Related Work

Literature reviews about frequent pattern mining and high utility mining are given in this section.

2.1 Frequent pattern mining

Extensive studies have been proposed for finding frequent patterns in transaction databases [8], [10]. Frequent itemset mining [5], [9] is the most popular topic among them. Apriori [9] is the pioneer for mining frequent itemsets from transaction databases by a level-wise candidate generation-and-test method. Tree-based algorithms such as FP-Growth[5] were proposed later to improve frequent itemset mining. FP-Growth improves the efficiency of frequent itemset mining since it does not have to generate candidate itemsets during the mining process and it only scans the database twice.

2. 2 High utility pattern mining

In frequent pattern mining field, past researches consider the importance of items uniformly. Thus, a new topic is raised for conquering this problem, that is, utility mining [11],[12],[13],[16]. In utility mining, each item may have different importance, such as profits and degree of user interest. The importance is generally called *utility*. Chan et al. first proposed the problem of utility mining in [13]. Yao et al. proposed the UMining algorithm [3] by applying an estimation method to prune the search space. However, it cannot capture the complete set of high utility itemsets since some high utility patterns may be pruned during the mining process. Among these researches, Liu et al. [16] proposed the two-Phase algorithm, which uses the transaction-weighted downward closure property to maintain downward closure property in utility mining. Although Two-Phase algorithm can reduce the search space of utility mining, it still generates too many candidates. Thus, proposed an isolated items discarding strategy to reduce the number of candidates by pruning isolated items during the level-wise searches. Transaction weighted utility model is efficient in terms of (1) Fewer candidates set (2) Accuracy and (3) Less arithmetic complexity compared to UMining and Umining_H[1]. This algorithm suffers from the same problem of level-wise candidate generation-and-test methodology then proposed [17] CTU-mine algorithm for mining high utility itemsets using pattern growth approach. In this tree, each node keeps its quantities and prefix related information separately and the author claims that the algorithm works more efficiently than Two-Phase for long and dense datasets when utility threshold is very low.

3. Terms and Definitions

The basic terms and formal definition of high utility itemset mining based on [11][12] and related concepts are described below.

Let I = {i1, i2, i3… im} be a set of items. An itemset X is nonempty subset of I. TDB = {T1, T2, T3….Tn} is a transactional database. Each transaction Ti is a set of items and subset of I. The local quantity of an item ip in a transaction Tq is denoted by l (ip, Tq), is defined as sales quantities stored in the transaction. The external utility e(ip) is the profit value per unit of item ip in the profit table. The utility mining problem is to discover all itemsets in a transaction database D with utility values higher than the minimum utility threshold.

Table 1. Transaction Table

TID	Transactions	Transaction Utility(tu)	Assigned Slave Node
T1	B(3), C(2), D(3)	59	P0
T2	A(3), D(2), E(2)	42	
T3	B(3), E(2)	40	
T4	A(1), B(1), C(1)	20	
T5	A(2), B(3,), D?(5)	77	P1
T6	A(3), B(4)	58	
T7	E(1)	5	
T8	B(2), D(2)	34	

Table 2. External Utility

Item	Utility
A	6
B	10
C	4
D	7
E	5

Definition 1: The **utility** of item ip in transaction Tq, is the quantity measure denoted by U (ip, Tq), Where

$$\mathbf{U(ip, Tq) = l(ip, Tq) \times e(ip)}$$

Definition 2: The **utility value** of an itemset X in the database U(X), is given as

$$\mathbf{U(X) = \sum_{ip\ \varepsilon\ X} \sum_{Tq\ \varepsilon\ D} U(ip, Tq)}$$

Definition 3: The **transaction utility** of transaction Tq, denoted as tu (Tq), is the sum of the total profit of all items in Tq and it is defined by,

$$\mathbf{tu(Tq) = \sum_{ip\ \varepsilon\ Tq} U(ip, Tq)}$$

The last column of Table 1(a) gives the transaction utility of each transaction.

Definition 4: The **minimum utility threshold** is the user preferred percentile of total transaction utility value of the given database.

$$\mathbf{min\_util = \partial \; X \; \sum_{Tq\ \varepsilon\ D} tu(Tq)}$$

where ∂ is the user preferred percentage.

Definition 5: Local transaction utility utilization of an itemset X, denoted by ltwu(X), is the sum of the Transaction utilities of all transactions containing X in particular node is defined by,

$$\mathbf{Itwu(X) = \sum_{X \zeta Tq\ \varepsilon\ D} tu(Tq)}$$

Where X ζTq means X is subset of Tq.

Definition 6: Global transaction utility utilization of an itemset X, denoted by gtwu(X), is the sum of the transaction utilities of all transactions of all the nodes that containing X and defined by,

$$\mathbf{gtwu(X) = \sum_{i=1}^{i=p} \sum_{X \zeta Tq\ \varepsilon\ D} tui(Tq)}$$

Where X ζ Tq means X is subset of Tq

4. Proposed Framework and Algorithm

4.1 Framework

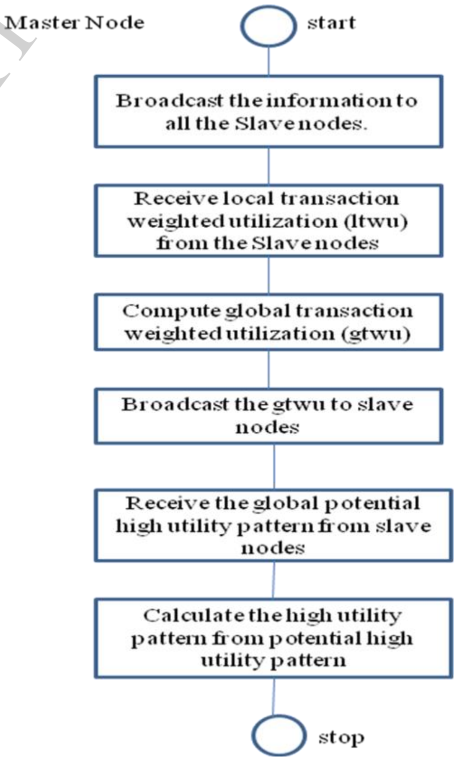
Master Node
start
Broadcast the information to all the Slave nodes.
Receive local transaction weighted utilization (ltwu) from the Slave nodes
Compute global transaction weighted utilization (gtwu)
Broadcast the gtwu to slave nodes
Receive the global potential high utility pattern from slave nodes
Calculate the high utility pattern from potential high utility pattern
stop

Figure1. Framework of Master Node

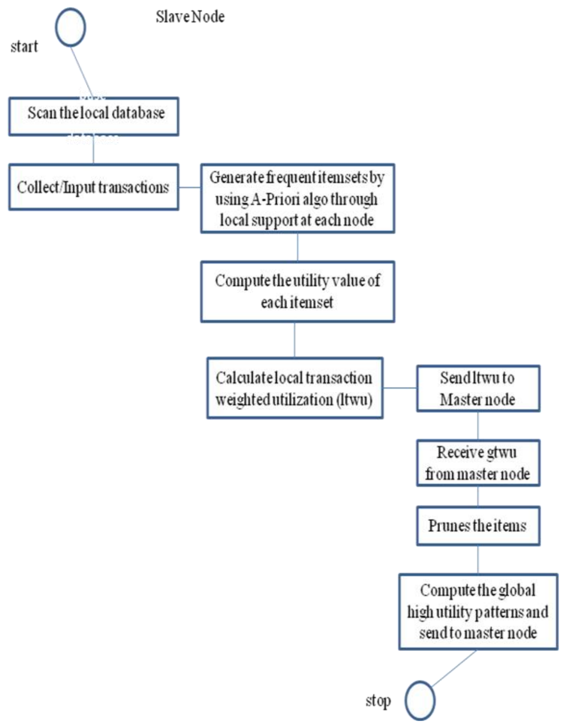
Slave Node
start
Scan the local database
Collect/Input transactions
Generate frequent itemsets by using A-Priori algo through local support at each node
Compute the utility value of each itemset
Calculate local transaction weighted utilization (ltwu)
Send ltwu to Master node
Receive gtwu from master node
Prunes the items
Compute the global high utility patterns and send to master node
stop

Figure2. Framework of Slave Node

4.2 The proposed algorithm works in the following steps:

Input : A Product Database
Output: All high utility Itemsets
Method:
1. Distribute large database from master node to all the Slave nodes.
2. Each slave node scan local database.
3. Start selecting items from the datasets.
   a. Select multiple items
   b. Add them to the transaction list
   c. Maintain a transaction dictionary of the transaction id and the items chosen for that transaction.
   d. Loop
4. Enter the minimum local support and confidence.
5. The frequent itemsets are generated using the A-priori Algorithm.
6. Compute the utility value of each itemset.
7. Calculate the local transaction weighted utilization (ltwu) of each node.
8. Send (ltwu) to the Master node.
9. Master node compute global transaction weighted utilization (gtwu).
10. Broadcast the gtwu to all slave nodes.
11. Each local node builds their global transaction utility table using global transaction weighted utilization and prunes the items that do not satisfied the given threshold *min_util.*
12. At each slave node for each computed Utility value
      if utility(x) >= minimum utility threshold
        x is high utility itemset
        end if end for
13. Each slave node send these local potential high utility patterns to Master node.
14. At last, global high utility pattern are cached by the Master node.

4.3 Analysis of proposed algorithm

At first level, each slave node generates frequent itemsets from its local databases through A-Priori. At second level, every node calculates the local transaction-weighted utilization *(ltwu)* as the definition 5. Global transaction weighted utilization is calculated by Mater node using the definition 6 and is broadcasted to all slave nodes. Then each local node builds their global transaction utility table using global transaction- weighted -utilization and prunes the items that do not satisfied the given threshold *min_util* (∂). As for example, total transaction utility value is 335. If ∂ is 25% than the minimum utility value will be *min_util* = 0.25*335 = 83.75. gtwu(C) < min_util (∂), so "C" is pruned. After that each local node finds the potential high utility patterns that satisfied the given threshold min_util (∂). Each local node calculates the actual utility u(S) from potential high utility patterns by scanning the database and sends to Master node. Finally, global high utility pattern are accumulated by the Master node.

5. Conclusion

In this paper, a distributed method is proposed to generate complete set of high utility itemsets from large databases. It also prunes the low utility itemsets from transactions at initial level by using downward closure property. This approach creates distributed environment with one master node and some slave nodes. Large database is distributed to all salve nodes. At first level, each slave node generates frequent itemsets from its local databases through A-Priori. At second level, every node calculate local weighted utility, mine high utility itemsets and send it to Master node. Then master node calculate global weighted utility and find final global high utility patterns by accumulating local high utility patterns. So, the proposed method can provide the high scalability and performance gain and require minimum communication among the nodes. It can decrease the execution time by parallelizing pattern mining.

6. References

[1] R. Chithra, and S.Nickolas. “A Tree Based Novel Algorithm for High Utility Itemset Mining”, International Journal of Computational Intelligence Research, (2011).

[2] H. Yao, H.J. Hamilton, and C.J. Butz , “A Foundational approach to mining Itemset Utilities From Databases”, Third SIAM International Conference on Data Mining, (2004), pp. 482-486.

[3] H. Yao, and H.J. Hamilton, “Mining itemset utilities from Transactional databases, Data & Knowledge Engineering”, 59, 2006, pp. 603-626.

[4] R. Agarwal, C. Aggarwal, V.V.V. Prasad, “A tree projection algorithm for generation of Frequent itemsets”, Journal of Parallel and Distributed Computing, vol 61, (2001) 350–371.

[5] Han J, Pei J, Yin Y (2000) “Mining frequent patterns without candidate generation” In: Proc of the ACM-SIGMOD int’l conf on management of data, pp 1–12

[6] Pei J, Han J, Mortazavi-Asl B, Pinto H, Chen Q, Dayal U, Hsu MC (2004) “Mining sequential patterns by pattern-growth: the prefix span approach”. IEEE Trans Knowl Data Eng 16(10)

[7] Y. Liu, W.K. Liao and A. Choudhary (2005) “A Two Phase algorithm for fast Discovery of high utility itemset”, Cheng, D. And Liu, H. (eds) PAKDD, LNCS(LNAI), Vol 3518 pp 689- 695, Springer, Heidelberg.

[8] Adnan M, Alhajj R (2009) DR “FP-tree: disk-resident frequent pattern tree” Appl Intell 30(2):84–97

[9] Agrawal R, Srikant R (1994) “Fast algorithms for mining association rules” In: Proc. of the 20th int’l conf. on very large data bases, pp 487–499

[10] Agrawal R, Srikant R (1995) “Mining sequential patterns” In: Proc of 11th int’l conf on data mining, pp 3–14

[11] Ahmed C. F., Tanbeer S. K., Jeong B.-S., Lee Y.-Koo.: “Efficient Tree Structures for High Utility Pattern Mining in Incremental Databases” Transactions on Knowledge and Data Engineering, Vol. 21, No. 12, pp. 1708 – 1721 (2009)

[12] Ahmed CF, Tanbeer SK, Jeong B-S, Lee Y-K (2011) “HUC-Prune: an efficient candidate pruning technique to mine high utility patterns” Appl Intell 34(2):181–198

[13] Chan R, Yang Q, Shen Y (2003) “Mining high utility itemsets” In: Proc of third IEEE int’l conf on data mining, pp 19–26

[14] Tseng VS, Wu C-W, Shie B-E, Yu PS (2010) “UP-growth: an efficient algorithm for high utility itemsets mining” In: Proc of the 16th ACM SIGKDD conf on knowledge discovery and data mining (KDD’10), pp 253–262

[15] Yao H, Hamilton HJ (2006) “Mining itemset utilities from transaction databases” Data Knowl Eng 59:603–626

[16] Liu Y, Liao W-K, Choudhary A (2005) “A fast high utility itemsets mining algorithm” In: Proc of utility-based data mining

[17] A. Erwin, R.P. Gopalan, and N.R.Achuthan “CTU-Mine: An “Efficient High Utility Itemset Mining Algorithm Using the Pattern growth Approach,” Proc. Seventh IEEE Int’l Conf. Computer and Information Technology (CIT ’07), 2007,pp. 71- 76.