HIGH UTILITY IRREGULAR ITEMSETS MINING WITH MAP REDUCE

[1]Rafa. T.A, [2]Dr.M. Sharmila Kumari
[1]M.Tech Student, [2]Professor & Head of Department
Department of Computer Science & Engineering, P. A College of Engineering, Karnataka, India

Abstract: In Data mining, finding itemsets that gives high utilities based on high profit, low cost, low risk and other factors is termed as High-utility itemsets mining (HUIM). Extracting such information is useful in business and marketing for understanding buying pattern of customers. There are efforts to find high utility itemsets based on regular occurrence of itemsets, which is not useful in different applications. There can be some interested itemsets found where some products can give high profit even though customers do not regularly buy them together. It helps to improve sale amount of profit and marketing strategy. So this paper aims to find high utility irregular itemsets (HUII) using map-reduce in order to perform task in parallel and improve performance.

***Index Terms* - High-utility itemsets mining (HUIM), high utility irregular itemsets (HUII), map reduce.**

I. INTRODUCTION

Association rule mining (ARM) and frequent itemsets mining (FIM) are major theories in data mining and they have several applications. Yet traditional FIM considers only whether an item occurs in a transaction or not. It does not consider the significance of these items which can be used in different applications that can discover interesting itemsets. To avoid these issues Chan et al. [1] proposed to determine interesting itemsets by considering unit utility of each item and its number of occurrence in transaction. Thus High utility Itemsets mining (HUIM) is announced. But they are not sufficient for some applications. For example, LCD TV and smart digital box can give great profit even though customers do not buy them regularly and together. So inorder to avoid decline of such products and layout shelf effectively, new method is proposed in this project.

Map reduce is a programming model enables to process and generates large quantity of data in parallel, distributed environment. The input data is divided and placed in different nodes. Map reduce consists of two functions: map and reduce.
Map Function: The input to map function is a <key, value> pair and output is a list of <key, value> pair in different domain.
Map function is executed in parallel to every group of input data sets. Reduce Function: The input to reduce function is the sorted output of the map function which is <key, list of values>and its output is a collection of values.

II. PROBLEM DEFINITION

Let I = {i1, i2, . . . , im} be the set of m items. Every item ij ∈ I has particular unit utility stated in terms of profit, cost, risk and other user-defined factors, called external utility, EU(ij). This is shown in Table 2.1.

Table 2.1: Unit Profits associated with item

Item	External Utility
A	2
B	3
C	4
D	20
E	2
F	25
G	5
H	3

A set Y = {ij,...,ik}⊆I is called a k-itemset, if Y contains k items. A transactional database D = {t1, t2,...,tr} has set of r transactions in which each transaction tp ∈ D has transaction identifier p (called TID) and set of items. Each item has quantity of its own occurrence in transaction called internal utility, IU(ij,tp).
Utility of an item is the multiplication of internal utility and external utility[2]
Ie. U(ij, tp)= IU(ij,tp) * EU(ij)
Transaction utility of a transaction tp is the summation of utility values of all items occurring in transaction tp, denoted as tu(tp)=∑ij∈tp U(ij,tp). It is shown in Table 2.2.

Table 2.2: Transaction Database

TID	Transaction items(internal utility)	Transaction Utility
T1	A(2), B(3), D(13), F(2)	323.0
T2	C(2), E(3),H(4)	26.0
T3	A(3), B(2)	12.0
T4	A(3), F(1), G(1)	36.0
T5	A(1),B(2), D(1)	28.0
T6	C(20)	80.0
T7	F(1),G(2), H(8)	59.0
T8	A(1), B(1), F(1),G(1)	35.0

The main challenge to mine high utility itemsets is downward closure property cannot be applied. That is if an itemset Y have a low utility value, we cannot ignore its superset as we don't know whether its supersets have low utility value or not.[3].
Map reduce is a programming model enables to process and generates large quantity of data in parallel, distributed environment. The input data is divided and placed in different nodes. Map reduce consists of two functions: map and reduce.
Map Function: The input to map function is a <key, value> pair and output is a list of <key, value> pair in different domain.
Map(k1, v1) -> list(k2, v2)
Map function is executed in parallel to every group of input data sets.
Reduce Function: The input to reduce function is the sorted output of the map function which is <key, list of values>and its output is a collection of values.
Reduce(k2, list (v2)) → list(v3)
Reduce function is executed in parallel to each group.

If an itemset Y has lower utility value, all of its superset cannot have high utility value, which helps to keep the downward closure property. The transaction weighted utility of an itemset Y in database D is the sum of transaction utilities of Y in D denoted as TWU(Y)=∑tp,tp∈D tu(tp).
if TWU(Y) is less than a give utility threshold (UT), all of Y's superset are not high utility items [4].
The tight over estimated utility of an itemset Y in database D is the sum of Y's utility and remaining utility of Y in database D, denoted as TOU(Y)=u(Y)+ru(Y).
Where remaining utility of itemset Y is the sum of all utilities of items in transaction ordered by TWU in ascending order after Y.
if TOU(Y) is less than a give utility threshold (TU), all itemsets ordered after Y are not high utility [5].

The regularity of itemset Y in two succeeding transactions containing Y in tp and tq where p < q is the gap of occurrence of Y between tp and tq denoted as R(Y,p,q) = q – p[6].
An itemset Y is called a high utility irregular itemset, if its utility value (U(Y)) is greater than or equal to utility threshold(UT)and its regularity(R(Y)) is greater than a user-given regularity threshold (RT).

Map reduce improves the performance of mining algorithms. Tasks is divided into sub tasks and given to homogenous nodes , so each node performs in parallel. Different map and reduce functions are run simultaneously and efficiency is highly improved[7].

III. METHODOLOGY

HUIIM contains three phases of map and reduce function. Figure 3.1 depicts overview of the process.

3.1 Phase 1:

The input to mapper in phase 1 is a key value pair where key is the transaction id and value is set of transaction items in that particular transaction. The output is also a <key, value> pair which contains transaction id and transaction items which is fed into the reducer. At reducer, scanning of database is done and creating EUL (Extended Utility List) structure which is described below.

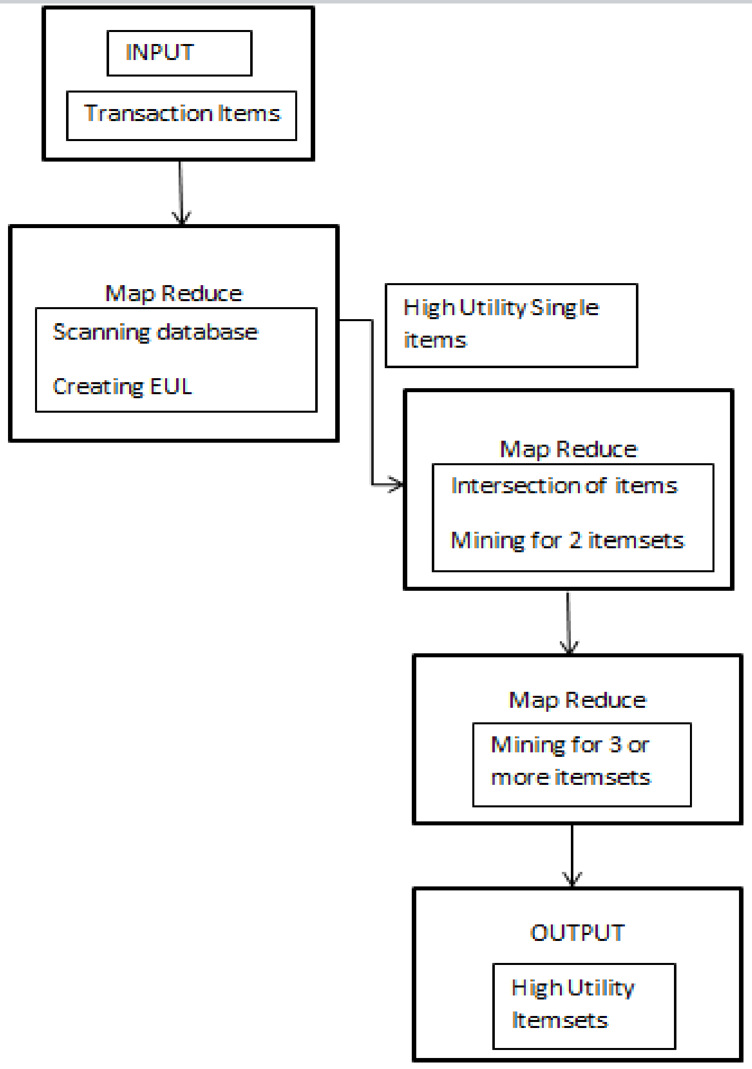

Fig 3.1 Overview of the process

Extended Utility List (EUL) Structure
This is an extension of utility list structure [4] used for keeping information about occurrence of each item/itemset along with utility values. For an itemset Y, its EUL can be represented as an ordered set of 4-tuples, denoted as EUL(Y) = {e1,e2,...,ek} where each entry ej = {TID,U(Y, Tp),RU(Y,Tp),UP(Y,Tp)} contains a) TID of transaction containing Y. b) utility of Y in transaction Tp. c) remaining utility of Y and d) utility prefix of Y in tp .This avoids to scan database again.

Scanning Database
A simple list structure called TU-List is used for storing transactional utility of every transaction in database. Every child node contains information about its own utility, remaining utility, transaction weighted utility, regularity, and EUL. Phase 1 map reduce algorithm is given below:

```
Map(Key, Transaction Itemsets)
Input is transaction id and transaction items
Step 1: Start
Step 2: For each transaction ti in database D,do
Step 3: Return transaction items
Step 4: End
Step 5: Stop
```

```
Reduce(key, transaction items)
Step 1: Start
Step 2: Create TUList to store transaction utilities of all transactions
Step 3: For each transaction ti in database D, do
Step 4: Calculate R(i)=max(R(i),p−q), q is TID of last occurrence of i (q is collected in the last entry in EUL(i)
Step 5: Calculate U(i,Tp) ← EU(i)×IU(i,tp) and collect an entry <p,u(i,tp),0,0> at tail of EUL(i)
Step 6: Calculate TWU(i) ← TWU(i)+TU(Tp)
Step 7: Collect the items whose utility is greater than UT and regularity greater than RT.
Step 8: Stop
```

3.2 Phase 2:

At the end of phase 1 of map reduce, single itemsets are obtained.

In phase 2 of high utility itemsets mining, different set of map reduce is used, where map function combines 2 different items to form itemsets , then calculate its transaction weighted utilities. Those items whose tight over utility (TOU) greater than utility threshold are combined with another item into 2 itemset Y. So EUL (i) and EUL (j) are combined to form EUL (ij) and all other parameters are calculated accordingly. If transaction weighted utility (TWU (ij)) is greater than utility threshold, item ij is considered as candidate itemset, and it is created as a child node of i. Finally at reducer, if utility of ij is greater than utility threshold and regularity of ij is greater than regularity threshold, then itemset ij is high utility irregular itemset.

```
Map (Key, Itemset)
Step 1: For every item i do
Step 2: If TOU (i) ≥ UT then
Step 3: For item j in transaction items (where i ≺ j) do
Step 4: EUL(ij) ←∅
Step 5: Intersect EUL (EUL(i), EUL(j))
Step 6: Compute R (ij), TWU(ij),U(ij),RU(ij), UP(ij) from EUL(ij)
Step 7: Compute TOU(ij) ← U(ij)+RU(ij)−UP(ij)
Step 8: If TWU (ij) ≥ UT then
Step 9: Create an itemset ij with PS(ij), TWU(ij), U(ij),RU(ij),UP(ij) and EUL(ij).
Step 10: End
```

```
Reduce (Key, Itemset)
Step 1: If U(ij) ≥ UT and R(ij) >RT then
Step 2: HUIIs ← HUIIs∪ij, Adding that ij to set of high utility irregular itemsets.
```

3.3 Phase 3

Phase 3 defines mining three or more itemsets, where each two itemset Y ={ij,ik} which has tight overestimated utility (TOU(Y) greater than utility threshold merge with another itemset Z= {ij,im}which has the same prefix as Y that is ij, then Y and Z can be combined to form 3 itemset {ij,ik,im}. Then EUL(Y) and EUL(Z) are combined to form EUL(YZ) and Compute R(YZ), TWU(YZ) U(YZ), RU(YZ), UP(YZ). If transaction weighted utility (TWU(YZ)) is greater than utility threshold, item YZ is considered as candidate itemset, and it is taken as itemset of Y. Finally if utility of YZ is greater than utility threshold and regularity of YZ is greater than regularity threshold, then itemset YZ is high utility irregular itemset.

```
Map (Key, Itemset)
Step 1: If TOU(Y ) ≥ UT then
Step 2: Let lY be the last item in itemset Y.
Step 3: Let lZ be the last item in itemset Z.
Step 4: Collect the prefix itemset.
Step 5: End
```

```
Reduce (Key, Itemset)
Step 1: Intersect EUL (EUL(YZ), EUL(Y) , EUL(Z))
Step 2: Compute R(YZ), TWU(YZ) U(YZ), RU(YZ), UP(YZ) from EUL(YZ)
Step 3: Assign TOU(YZ) = U(YZ)+RU(YZ)−UP(YZ)
Step 4: If TWU(ij) ≥ UT then
Step 5: create an itemset YZ with PS(YZ), TWU(YZ), U(YZ), RU(YZ), UP(YZ) and EUL(YZ)
Step 6: if U(YZ) ≥ UT and R(YZ) >RT
Step 7: HUIIs ← HUIIs∪YZ
```

IV. RESULTS

Table 4.1 displayed high utility itemsets with irregular occurrence, its utility, regularity, TWU and remaining utility, if utility threshold is taken as 40.0 and regularity is taken as 3.

Table 4.1: Result set of High Utility Irregular Itemsets Mining

Sl.No.	Itemset	Utility	Regularity	TWU	Remaining Utility
1	C	88.0	4	106.0	0.0
2	D	280.0	4	351.0	71.0
3	AD	286.0	4	351.0	50.0
4	BD	295.0	4	351.0	56.0
5	DF	310.0	4	323.0	0.0
6	FG	95.0	4	130.0	0.0
7	FH	49.0	5	59.0	0.0
8	ABD	292.0	4	351.0	50.0
9	ADF	314.0	4	323.0	0.0
10	AFG	68.0	4	71.0	9.0
11	BDF	68.0	4	323.0	0.0
12	FGH	74.0	5	59.0	0.0
13	ABDF	313.0	4	323.0	0.0

IV. CONCLUSION

In this paper we have discovered high utility irregular itemsets, where we got the products set that gives high profit even though customers are not frequently purchase them together. It helps to build marketing strategy, manage warehouse etc. The use of map reduce helps in parallel processing of data and finds the solution easily when there is large number of data. Its ability to scale is also a major advantage which improves performance of program. The ideas of transaction weighted utility, remaining utility and tight overestimated utility are used to filter unwanted itemsets. It also used EUL (Extended Utility List structure) to store regularity information along with utility value of every itemset. Experiments were done to prove that map reduce based high utility irregular itemsets mining is runtime and memory efficient.

REFERENCES

[1] R. Chan, Q. Yang, and Y.-D. Shen, "Mining high utility itemsets," in Data Mining, 2003. ICDM 2003. Third IEEE International Conference on, 2003, pp. 19–26.

[2] Shalini Zanzote Ninoria, S. S. Thakur, "A Survey on High Utility Itemsets Mining", International Journal of Computer Applications (0975 – 8887) Volume 175 – No.4, October 2017

[3] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules in large databases," in VLDB, 1994, pp. 487–499.

[4] Y. Liu, W.-k. Liao, and A. Choudhary, "A two-phase algorithm for fast discovery of high utility itemsets," in Advances in Knowledge Discovery and Data Mining, 2005, vol. 3518, pp. 689–695.

[5] M. Liu and J. Qu, "Mining high utility itemsets without candidate generation," in Proceedings of the 21st ACM International Conference on Information and Knowledge Management, 2012, pp. 55–64.

[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 Transactions on Knowledge and Data Engineering, vol. 25, no. 8, pp. 1772–1786, 2013.

[7] Ming-Yen Lin, Pei-Yu Lee, Sue-Chen Hsueh, "Apriori-based Frequent Itemset Mining Algorithms on Map Reduce".

[8] P. Fournier-Viger, C.-W. Wu, S. Zida, and V. S. Tseng, FHM: Faster High-Utility Itemset Mining Using Estimated Utility Co-occurrence Pruning, 2014, pp. 83–92. [7] S. Zida, P. Fournier-Viger, J. C.-W. Lin, C.-W. Wu, and V. S. Tseng, "Efim: a fast and memory efficient algorithm for high-utility itemset mining," Knowledge and Information Systems, pp. 1–31, 2016.

[9] J. Liu, K. Wang, and B. C. M. Fung, "Mining high utility patterns in one phase without generating candidates," IEEE Transactions on Knowledge and Data Engineering, vol. 28, no. 5, pp. 1245–1257, 2016.

[10] K. Amphawan and A. Surarerks, "Pushing regularity constraint on high utility itemsets mining, "in Advanced Informatics: Concepts, Theory and Applications, 2015 2nd International Conference on, 2015, pp. 1–6.

[11] P. Fournier-Viger, J. C.-W. Lin, Q.-H. Duong, and T.-L. Dam, PHM: Mining Periodic High-Utility Itemsets, 2016, pp. 64–79.

[12] K. Amphawan, P. Lenca, A. Jitpattanakul, and A. Surarerks, “Mining high utility itemsets with regular occurrence,” Journal of ICT Research and Applications, vol. 10, no. 2, pp. 153–176, 2016.
[13] P. F. Viger, “SPMF: An Open-Source Data Mining Library,” http://www.philippe-fournier-viger.com/spmf/, 2015.
[14] S. K. Tanbeer, C. F. Ahmed, B.-S. Jeong, and Y.-K. Lee, “Discovering periodic-frequent patterns in transactional databases,” in Proceedings of the 13th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining, 2009, pp. 242–253.
[15] K. Amphawan, P. Lenca, and A. Surarerks, “Mining top-k periodic frequent patterns without support threshold,” in Proceedings of the 3rd International Conference on Advances in Information Technology, vol. 55, 2009, pp. 18–29.