IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Volume-2, Issue-9 E-ISSN: 2347-2693

Efficient Approach for Mining High Utility Itemsets From Transactional Databases

P.S.Prakash[1*] and S.Anandamurugan[2]

[1*,2]Department of Information Technology, Kongu Engineering College, Perundurai, Erode, India

www.ijcaonline.org

Received: Aug/21`/2014 Revised: Sep/09/2014 Accepted: Sep/20/2014 Published: Sep/30/2014

Abstract— Mining high utility item sets from transactional database refers to the discovery of item sets with high utility like profits. Although a number of relevant algorithms have been proposed in recent years, they incur the problem of producing a large number of candidate item sets for high utility item sets. Such a large number of candidate item sets degrades the mining performance in terms of execution time and space requirement. The situation may become worse when the database contains lots of long transactions or long high utility item sets. An algorithm, namely UP-Growth proposed [7] for mining high utility item sets with a set of effective strategies for pruning candidate item sets. The information of high utility item sets is maintained in a tree-based data structure named UP-Tree such that candidate item sets can be generated efficiently with only two scans of database. Experimental results show that the proposed algorithm, especially UP-Growth, not only reduce the number of candidates effectively but also outperform other algorithms substantially in terms of runtime, especially when databases contain more no of transactions.

Keywords—High Utility Itemsets, Datamining, Transactional Database

I. INTRODUCTION

1.1 DATA MINING

Data Mining is the process of analyzing data from different perspectives and summarizing it into useful information. Simply, data mining refers to extracting or mining knowledge from large amounts of data. Other terms of data mining are knowledge mining from data, knowledge extraction, data/pattern analysis, data archaeology, and data dredging.

Data mining functionalities include:

Association	-looking for patterns where one event is connected to another event
Classification	- looking for new patterns
Clustering	-finding and visually documenting groups of facts not previously known
Prediction	-predicting missing or unavailable numerical data values rather than class labels

1.2 UTILITY MINING

Mining high utility itemsets from Transactional databases refers to finding the itemsets with high profit .Utility of an items in transactional Databases Consists of two aspects.

1.External utility- Importance of distinct items is called external utility or unit profit value.

2.Internal utility- Importance of items in transactions is called internal utility or support count.

Corresponding Author: *P.S.Prakash*

1.3 UTILITY OF AN ITEMSET

Utility of an itemset is defined as product of its External Utility and its Internal Utility or multiplication of unit profit value of the particular item and its support count.

1.3.1 High Utility itemset

An itemset is called a high utility itemset if its utility is no less than a user specified minimum utility threshold.High utilityitemset is also called as promising itemset which is used for discovering PHUIs from the transactional database.

1.3.2 Low Utility itemset

An itemset is called a low utility itemset if its utility is less than a user specified minimum utility threshold. Low utility itemset is also called as unpromising itemset which contain unprofitable items. So, this is not used for discovering PHUIs from the transactional database.

1.4 UP-GROWTH

UP-Growth is an algorithm for discovering high utility itemsets and maintaining important information related to utility patterns within databases by using UP-Tree and user specified minimum utility value.

II. LITERATURE REVIEW

2.1 RELATED WORK

Pattern growth based association rule mining algorithms such as FP-Growth were proposed in the paper [5]. It finds frequent itemsets without generating any candidate itemset and scans database just twice.

In frequent itemset mining or FP-Growth the importance of items to users is not considered.so the concept of weighted items was first proposed.[3].This concept does not have downward closure property and hencemining performance cannot be improved.

To overcome this problem the concept of Weighted downward closure property was proposed.[2]. By using transaction weight, weighted support cannot reflect only the importance of an itemset but also maintain the downward closure property during the mining process but transaction databases, items' quantities in transactions are not taken into consideration.

Two-Phase algorithm was proposed in the paper [6].It is mainly composed of two mining phases. Inphase I and phase II, it employs an Apriori-based level-wise method to enumerate HTWUIs and HTWUIs that are high utility itemsets are identified with an additional database scan.Although Two-Phase algorithm reduces search space by using TWDC property, it still generates too many candidates to obtain HTWUIs and requires multiple database scans.

To overcome this problem, Li et al.proposed an Isolated Items Discarding strategy or IIDS to reduce the number of candidates. By pruning isolated items during level-wise search, the number of candidate itemsets for HTWUIs in phase I can be reduced but this algorithm still scans database for several times and uses a candidate generation-and-test scheme to find high utility itemsets.

To efficiently generate HTWUIs and avoid scanning database too many times, IHUP algorithm was proposed in the paper [1]. A tree based structure called IHUP-Tree is used to maintain the information about itemsets and their utilities.

2.2 DRAWBACKS IN EXISTING SYSTEM

IHUP algorithm produce too many HTWUIs inphase I since the overestimated utility calculated by TWUis too large. Such a large number of HTWUIs will degrade the mining performance in phase I substantially in termsof execution time and memory consumption and also if requires too many database scans for discovering the high utility itemsets.

2.3 PROBLEM DEFINITION

Given a transaction database D and a user-specified minimum utility threshold min_util,the problem of mining high utility itemsets from D is tofind the complete set of the itemsets whose utilities aregreater than or equal to min_util.

2.4 PROPOSED SYSTEM

To facilitate the mining performance and avoid scanning original database repeatedly, we use a compact tree structure, named UP-Tree.To maintain the information of transactions and high utility itemsets, four strategies are applied to minimize the overestimated utilities stored in the nodes of global UP-Tree.UP-Growth algorithm consistsof three steps:

- Scan the database twice to construct a global UP-Tree with the strategy DGU and DGN.
- Recursively generate potentialhigh utility itemsetsfrom global and local
- UP-Tree by UP-Growth with the strategy DLU and DLN.
- Identify actual high utility itemsets from the set of PHUIs

III. SYSTEM IMPLEMENTATION

The main objective of "Mining High Utility Item sets from Transactional Databases" is to enhance the mining performance in terms of Execution time and Space requirement by using UP- Growth Algorithm. There are several modules involved in finding high utility item sets from transactional databases. They are as follows

- Data Collection
- Calculation of TU and TWU Value
- Construction of RTU and UP-Tree.
- Construction of CPB for each item in
- Reorganized Transaction table and finding high utility item set

3.1 DATA COLLECTION

The data has been collected from an expert person who published the paper called A FAST ALGORITHM FOR MINING HIGH UTILITY ITEM SETS. We request a retail store data set to that person through e-mail. He sent that retail store data set for us. This dataset is composed of an items, a Transaction IDs, Transaction details of the items and unit profit value of each items. Totally 2000 items are obtained from that retail store data set.

3.2 *CALCULATION OF TU AND TWU VALUE*

A transaction database D = {T1, T2, …,Tn} contains a set of transactions, and each transaction Td (1 <d <n) has aunique identifier d, called TID. Each item ipin transactionTd is associated with a quantity q(ip, Td), that is, the purchase quantity of ipin Td.

3.2.1 Formula for CalculatingTU Value

1. Utility of an item ip in a transaction Td is denoted as u(ip, Td) and defined as pr(ip) × q(ip, Td).
2. Utility of an itemset X in Td is denoted as u(X, Td) and defined as

$$\sum_{i_p \in X \wedge X \subseteq T_d} u(i_p, T_d)$$

3.2.2 Formula for CalculatingTWU Value

Transaction-weighted utility of an itemsetX is the sum of the transaction utilities of all the transactionscontaining X, which is denoted as TWU(X) and defined as

$$\sum_{X \subseteq T_d \wedge T_d \in D} \mathrm{TU}(T_d)$$

3.3 CONSTRUCTION OF RTU AND UP-TREE

Two strategies are used for decreasing the overestimated utility of each item during the construction of Reorganized Transaction Table and a global UP-Tree.

3.3.1Construction of RTT

Strategy 1: DGU

Discarding Global Unpromising items and their actual utilities from transactions and transaction utilities of the database. The transactions are reorganized by pruning the unpromising items and sorting the remaining promising items in a fixed order. Any ordering can be used such as the lexicographic, support or TWU order. Each transaction after the above reorganization is called a reorganized transaction. New TU after pruning unpromising items is called reorganized transaction utility denoted as RTU.Reorganized Transaction table is created by using this new TU value and the items. That the subroutine of Insert_Reorganized_Transaction is given below. Reorganized Transaction Table is shown in Table1 below

Table1: Reorganized Transactions and their RTUs

TID	Reorganized transactions	RTU
T_1	(C,10) (D,1) (A,1)	17
T_2	(E,2) (C,6) (A,2)	22
T_3	(E,2) (D,6) (A,2)(B,2)	32
T_4	(E,1) (C,13) (D,2)(B,4)	30
T_5	(E,1) (C,4) (B,2)	11
T_6	(C,1) (D,1) (A,1)(B,1)	10

3.3.2 Construction of UP-TREE

Strategy 2: DGN

Decreasing Global Node utilities for the nodes of global UP-Tree by actual utilities of descendant nodes during the construction of global UP-Tree.

By applying strategy DGN, the utilities of the nodes that are closer to the root of a global UP-Tree are further reduced.DGN is especially suitable for the databases containing lots of long transactions. A function

```
Subroutine: $Insert\_Reorganized\_Transaction(\mathrm{N}, i_x)$
Line 1: If N has a child $\mathrm{N}_{i_x}$ such that $\mathrm{N}_{i_x}.item = i_x$, increment $\mathrm{N}_{i_x}.count$
        by 1. Otherwise, create a new child node $\mathrm{N}_{i_x}$ with $\mathrm{N}_{i_x}.item = i_x$,
        $\mathrm{N}_{i_x}.count = 1$, $\mathrm{N}_{i_x}.parent = \mathrm{N}$ and $\mathrm{N}_{i_x}.nu = 0$.
Line 2: Increase $\mathrm{N}_{i_x}.nu$ by $(\mathrm{RTU}(t_j') - \sum_{p=x+1}^{n} u(i_p, t_j'))$, where $i_p \in t_j'$.
Line 3: If $x \neq n$, call $Insert\_Reorganized\_Transaction(\mathrm{N}_{i_x}, i_{x+1})$
```

Insert_Reorganized_Transactionis calledto applies DGN during the construction of a global UP-Tree. Its subroutine is given below

UP-Tree is constructed by two steps:

1. Finding Promising items
2. Constructing UP-Tree

1. Finding Promising items

Promising items are found by comparing TWU values of each items to the minimum utility threshold value. After sorting the promising items in the descending order header table was constructed. A table named header table is employed to facilitate the traversal of UP-Tree. In header table, each entry records an item name, an overestimated utility, and a link. The link points to the last occurrence of the node which hasthe same item as the entry in the UP-Tree. By following the links in header table and the nodes in UP-Tree, the nodes having the same name can be traversed efficiently.

2. Constructing UP-Tree

UP-Tree is constructed using reorganized transactions with header table. An algorithm for constructing UP-Tree is given below.Example of UP-Tree is shown in Figure 1.

```
Subroutine: UP-Growth(TX, HX, X)
Input: A UP-Tree TX, a header table HX for TX, an item setX, and a minimum utility threshold min_util.
Output: All PHUIs in TX.
(1) For each entry ikin HX do
(2) Trace each node related to ikvia ik.hlinkand accumulate ik.nutonusum(ik) ;
/* nusum(ik): the sum of node utilities of ik*/
(3) If nusum(ik)≥min_util, do
(4) Generate a PHUI Y = X ∩ ik;
(5) Set pu(ik) as estimated utility of Y;
(6) Construct Y-CPB;
(7) Put local promising items in Y-CPB into HY
(8) Apply DLU to reduce path utilities of the paths;
(9) Apply Insert_Reorgnized_Pathto insert paths into TY with DLN;
(10) If TY ≠null then call UP-Growth(TY, HY, Y);
(11) End if
(12)End for
```

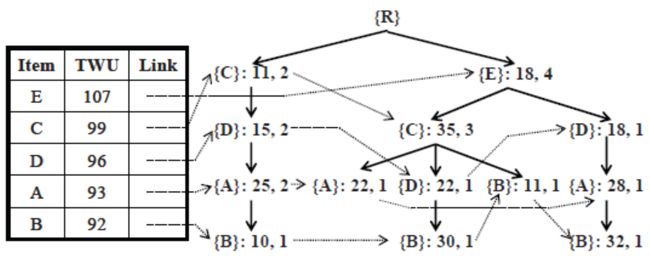


Figure 1: A UP-Tree by applying strategies DGU and DGN.

3.4 CONSTRUCTION OF CPB FOR EACH ITEM IN THE REORGANIZED TRANSACTION TABLE ANDGENERATING PHUIs

A basic method for generating PHUIs is to mine UP-Tree by UP-Growth. In this DLU and DLN strategies are used. By the strategies, overestimated utilities of item sets can be decreased and thus the number of PHUIs can be further reduced.
The common method for generating patterns in tree based algorithms contains three steps:
1.Generate conditional pattern bases by tracing the paths in the original tree
2.Construct conditional trees or local trees by the information in conditional pattern bases
3.Mine patterns from the conditional trees.

1. Generate conditional pattern bases by tracing the paths in the original tree

Strategy 3: DLU
Discarding Local Unpromising items and their estimated utilities from the paths and path utilities of conditional pattern bases by Equation 1

$$pu(p,\{i_m\}-CPB) = N_{i_m}.nu - \sum_{\forall i \in UI_{\{i_m\}-CPB} \wedge i \subseteq p} miu(i) \times p.count \quad (1),$$

Path utility of a path p in im's CPB is denoted as pu(p,{im}-CPB) and defined as Nim's node utility where p is retrieved by tracing Ni_m in the UP-Tree.
An example CPB for an item Band strategy 3 used for constructing the Reorganized path is shown in Table 4.2.Function for Insert Reorganized path is given below.

```
Subroutine : $Insert\_Reorganized\_Path(N, i_x)$
Line 1 : If N has a child $N_{i_x}$ such that $N_{i_x}.item = i_x$, increment $N_{i_x}.count$ by
    $p_j.count$. Otherwise, create a new child node $N_{i_x}$ with $N_{i_x}.item = i_x$,
    $N_{i_x}.count = p_j.count$, $N_{i_x}.parent = N$ and $N_{i_x}.nu = 0$.
Line 2 : Increase $N_{i_x}.nu$ by Eq (3).
Line 3 : If there exists a node $N_{i_x}$ in $p_j$ where $x+1 < m'$,
    call $Insert\_Reorganized\_Path(N_{i_x}, i_{x+1})$
```

Table 2: Retrived path and Reorganized path

Retrieved Path: Path utility	Reorganized path: Path utility (after DLU)
<ADC>:10	**<DC>:5**
<DCE>:30	**<EDC>:30**
<CE>:11	**<EC>:11**
<ADE>:32	**<ED>:27**

2. CONSTRUCT CONDITIONAL TREES OR LOCAL TREES BY THE INFORMATION IN CONDITIONAL PATTERN BASE

Strategy 4: DLN
Decreasing Local Node utilities for the nodes of local UP-Tree by estimated utilities of descendant nodes during the construction of global UP-Tree is performed by this formula

$$N_{i_k}.nu_{new} = N_{i_k}.nu_{old} + pu(p,\{i_m\}-CPB) - \sum_{j=k+1}^{m'} miu(i_j) \times p.count$$

Conditional tree for item B by using CPB-b after applying DLN is show in Figure2

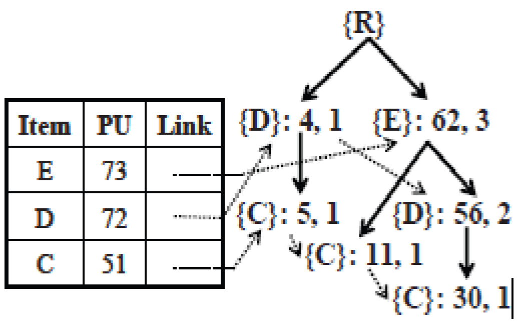


Figure 2: Conditional tree for CPB-b

3. MINE PATTERNS FROM CONDITIONAL TREES
Generating PHUIs from {B}-Tree by UP-Growth, the PHUIs that are involved with {B} are obtained, they are {B}:83, {BD}:60, {BDE}:56 and {BE}:62. After mining the remaining items in header table, all PHUIs in the UP-Tree in Figure 1 can be obtained, they are {A}:75, {B}:83, {BD}:60, {BDE}:56, {BE}:62 and {D}:55.

IV RESULT AND DISCUSSION

4.1 Construction of up-tree and CPB

UP-Growth algorithm is used to discover high utility items from the retail store dataset by user specified minimum utility threshold value and constructing UP-Tree based on the items, utility value and support count of the items. Here

strategy DGU and DGN are applied to construct UP-Tree. Strategy DLU and DLN are applied to construct CPB for the promising items. Various minimum utility threshold values are used and the results are analyzed. From this the minimum utility threshold value is found as 50 for discovering high utility items for retail store dataset.
From Retail store dataset, items, transactions of the items, unit profit value of each item are selected as input attributes to the UP-Growth algorithm. Totally eight items and six transactions are given as input to the algorithm and high utility item sets have been discovered.

4.2 EFFICIENTLY IDENTIFY HIGH UTILITY ITEMSETS

In IHUP algorithm generated number of HTWUIs is too large and scanning of original database consumes more time. But in the proposed algorithm called UP-Growth, overestimated utilities of PHUIs are smaller than or equal to TWUs of HTWUIs since they are reduced by the proposed strategies which are DGN, DGU, DLU, DLN. Thus, the number of PHUIs is much smaller than that of HTWUIs. High utility item sets are efficiently identified by using the UP-Growth algorithm.

4.2.1 Performance Comparison between IHUP and UP Growth

The number of candidate item sets generated by the IHUP algorithm is too large while comparing the IHUP and UP-Growth algorithm as shown in figure 3. So, IHUP algorithm is more time consuming when compared to the UP-Growth. UP-Growth uses only two database scans, one for constructing UP-Tree and another one for discovering PHUIs, because UP-Growth reduce the number of candidate item sets which is shown in figure 3.
In this figure3 different user specified minimum utility values are used to compare the IHUP and UP-Growth. Ten item sets were generated by UP-Growth algorithm for minimum utility threshold value fifty whereas eighteen item sets were generated by IHUP algorithm.

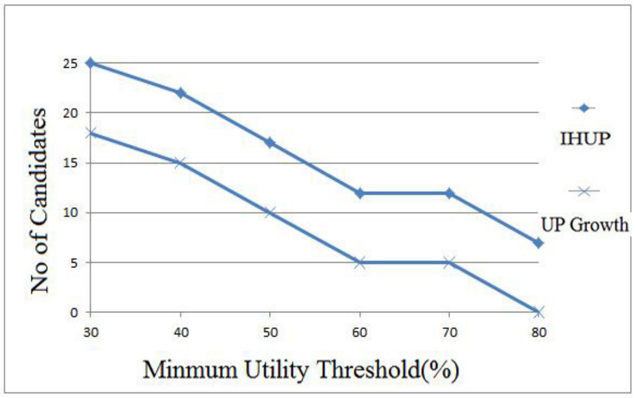


Figure 3: Number of candidates on Retail Store

From figure 3, it is found that the UP-Growth algorithm produces more efficient high utility item sets or candidates than other mining algorithms since it prunes the candidates based on the strategies and hence more effective than other algorithms.

V. CONCLUSION AND FUTURE WORK

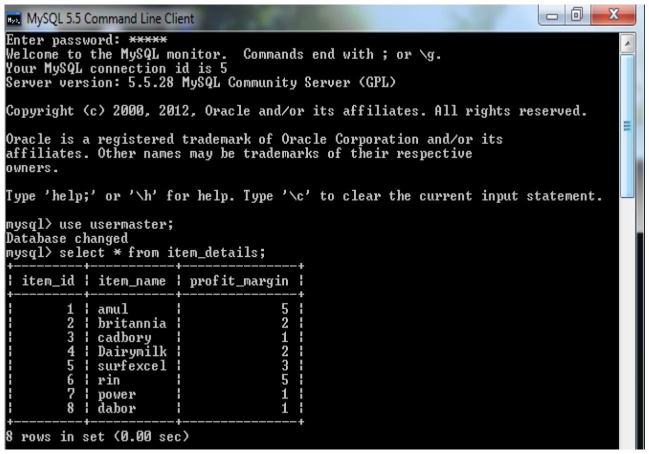
5.1 CONCLUSION

The Strategies named DGU, DGN, DLU, DLN are applied to decrease the overestimated utility value and the construction of Global UP-Tree and CPB for the items. A data structure named UP-Tree was proposed for maintaining the information of high utility item sets. Potential high utility item sets can be efficiently generated from UP-Tree with only two database scans. UP-Growth algorithm decrease overestimated utility and enhance the performance of utility mining in terms of execution time, space requirement and number of candidates.

5.2 FUTURE WORK

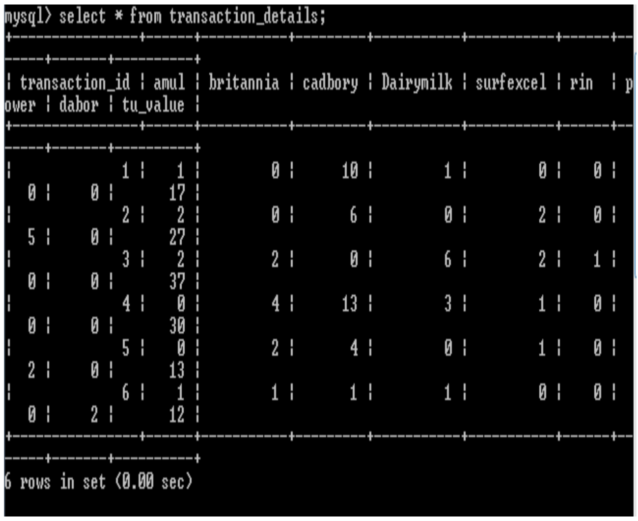
The proposed system deals only with retail store database. In future these retail store database can be extended to multiple shop databases such as jewellery shop, textile shopetc.

APPENDIX 1
SNAPSHOTS



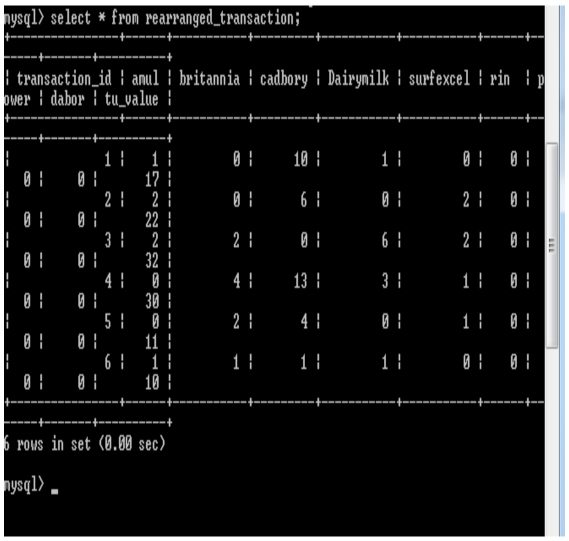
```
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 5
Server version: 5.5.28 MySQL Community Server (GPL)

Copyright (c) 2000, 2012, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use usermaster;
Database changed
mysql> select * from item_details;
+---------+-----------+---------------+
| item_id | item_name | profit_margin |
+---------+-----------+---------------+
|       1 | amul      |             5 |
|       2 | britannia |             2 |
|       3 | cadbory   |             1 |
|       4 | Dairymilk |             2 |
|       5 | surfexcel |             3 |
|       6 | rin       |             5 |
|       7 | power     |             1 |
|       8 | dabor     |             1 |
+---------+-----------+---------------+
8 rows in set (0.00 sec)
```

Figure A1.1: Item details



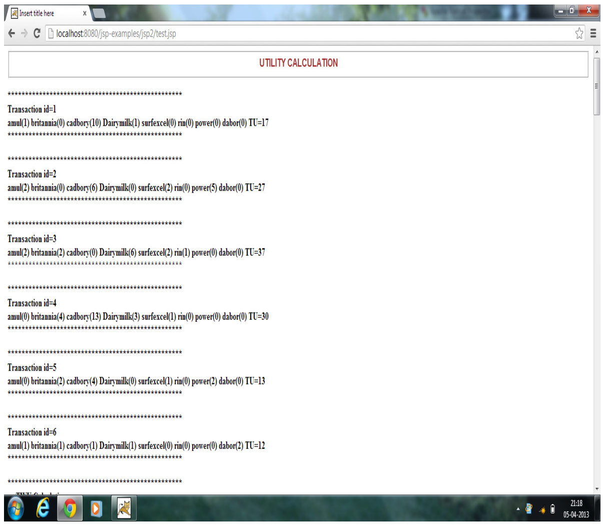
```
mysql> select * from transaction_details;
+----------------+------+-----------+---------+-----------+-----------+-----+--
-----+-------+----------+
| transaction_id | amul | britannia | cadbory | Dairymilk | surfexcel | rin | p
ower | dabor | tu_value |
+----------------+------+-----------+---------+-----------+-----------+-----+--
-----+-------+----------+
|              1 |    1 |         0 |      10 |         1 |         0 |   0 |
    0 |     0 |       17 |
|              2 |    2 |         0 |       6 |         0 |         2 |   0 |
    5 |     0 |       27 |
|              3 |    2 |         2 |       0 |         6 |         2 |   1 |
    0 |     0 |       37 |
|              4 |    0 |         4 |      13 |         3 |         1 |   0 |
    0 |     0 |       30 |
|              5 |    0 |         2 |       4 |         0 |         1 |   0 |
    2 |     0 |       13 |
|              6 |    1 |         1 |       1 |         1 |         0 |   0 |
    0 |     2 |       12 |
+----------------+------+-----------+---------+-----------+-----------+-----+--
-----+-------+----------+
6 rows in set (0.00 sec)
```

Figure A1.2: Transaction details



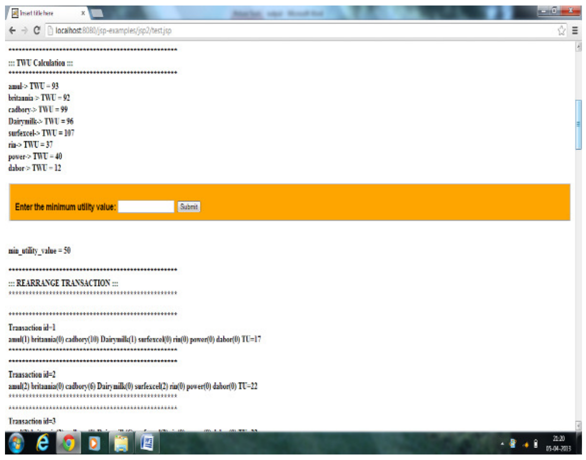
```
mysql> select * from rearranged_transaction;
+----------------+------+-----------+---------+-----------+-----------+-----+--
-----+-------+----------+
| transaction_id | amul | britannia | cadbory | Dairymilk | surfexcel | rin | p
ower | dabor | tu_value |
+----------------+------+-----------+---------+-----------+-----------+-----+--
-----+-------+----------+
|              1 |    1 |         0 |      10 |         1 |         0 |   0 |
   0 |     0 |       17 |
|              2 |    2 |         0 |       6 |         0 |         2 |   0 |
   0 |     0 |       22 |
|              3 |    2 |         2 |       0 |         6 |         2 |   0 |
   0 |     0 |       32 |
|              4 |    0 |         4 |      13 |         3 |         1 |   0 |
   0 |     0 |       30 |
|              5 |    0 |         2 |       4 |         0 |         1 |   0 |
   0 |     0 |       11 |
|              6 |    1 |         1 |       1 |         1 |         0 |   0 |
   0 |     0 |       10 |
+----------------+------+-----------+---------+-----------+-----------+-----+--
-----+-------+----------+
6 rows in set (0.00 sec)

mysql> _
```

Figure A1.3: Rearranged Transactions



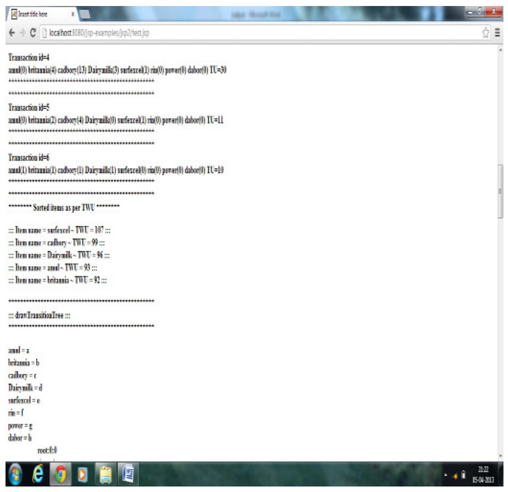
```
UTILITY CALCULATION

***************************************************
Transaction id=1
amul(1) britannia(0) cadbory(10) Dairymilk(1) surfexcel(0) rin(0) power(0) dabor(0) TU=17
***************************************************

***************************************************
Transaction id=2
amul(2) britannia(0) cadbory(6) Dairymilk(0) surfexcel(2) rin(0) power(5) dabor(0) TU=27
***************************************************

***************************************************
Transaction id=3
amul(2) britannia(2) cadbory(0) Dairymilk(6) surfexcel(2) rin(1) power(0) dabor(0) TU=37
***************************************************

***************************************************
Transaction id=4
amul(0) britannia(4) cadbory(13) Dairymilk(3) surfexcel(1) rin(0) power(0) dabor(0) TU=30
***************************************************

***************************************************
Transaction id=5
amul(0) britannia(2) cadbory(4) Dairymilk(0) surfexcel(1) rin(0) power(2) dabor(0) TU=13
***************************************************

***************************************************
Transaction id=6
amul(1) britannia(1) cadbory(1) Dairymilk(1) surfexcel(0) rin(0) power(0) dabor(2) TU=12
***************************************************

***************************************************
```

Figure A1.4: Result after calculation of TU value



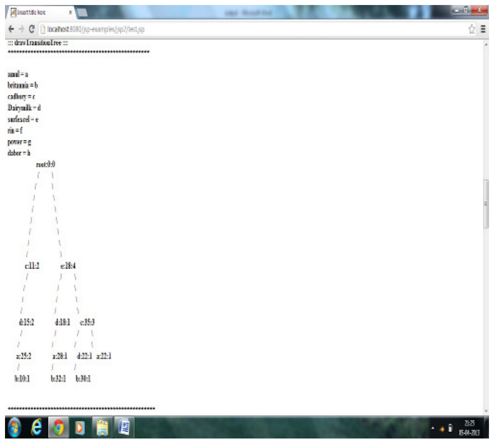
```
***************************************************
:::: TWU Calculation :::
***************************************************
amul-> TWU = 93
britannia-> TWU = 92
cadbory-> TWU = 99
Dairymilk-> TWU = 96
surfexcel-> TWU = 107
rin-> TWU = 37
power-> TWU = 40
dabor-> TWU = 12

Enter the minimum utility value: [          ] Submit

min_utility_value = 50

***************************************************
:::: REARRANGE TRANSACTION :::
***************************************************

***************************************************
Transaction id=1
amul(1) britannia(0) cadbory(10) Dairymilk(1) surfexcel(0) rin(0) power(0) dabor(0) TU=17
***************************************************
***************************************************
Transaction id=2
amul(2) britannia(0) cadbory(6) Dairymilk(0) surfexcel(2) rin(0) power(0) dabor(0) TU=22
***************************************************
***************************************************
Transaction id=3
```

Figure A1.5: Result after calculation of TWU value



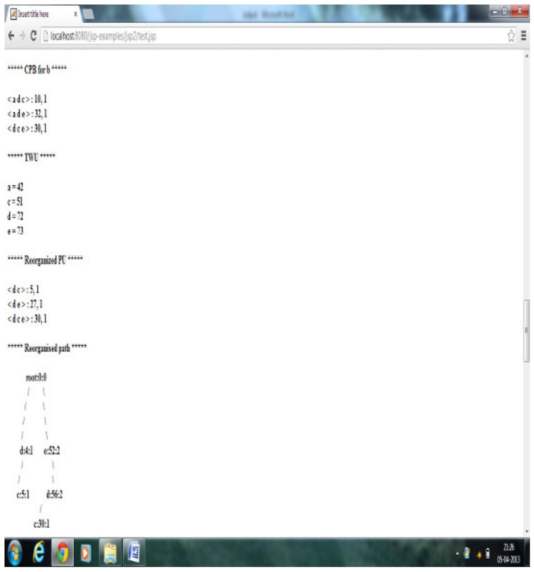
```
Transaction id=4
amul(0) britannia(4) cadbory(13) Dairymilk(3) surfexcel(1) rin(0) power(0) dabor(0) TU=30
***************************************************
***************************************************
Transaction id=5
amul(0) britannia(2) cadbory(4) Dairymilk(0) surfexcel(1) rin(0) power(0) dabor(0) TU=11
***************************************************
***************************************************
Transaction id=6
amul(1) britannia(1) cadbory(1) Dairymilk(1) surfexcel(0) rin(0) power(0) dabor(0) TU=10
***************************************************
***************************************************
******** Sorted items as per TWU ********

::: Item name = surfexcel ~ TWU = 107 :::
::: Item name = cadbory ~ TWU = 99 :::
::: Item name = Dairymilk ~ TWU = 96 :::
::: Item name = amul ~ TWU = 93 :::
::: Item name = britannia ~ TWU = 92 :::

***************************************************
::: drawTransitionTree :::
***************************************************

amul = a
britannia = b
cadbory = c
Dairymilk = d
surfexcel = e
rin = f
power = g
dabor = h
             root:0:0
```

Figure A1.6: Result after Rejection of unpromising items



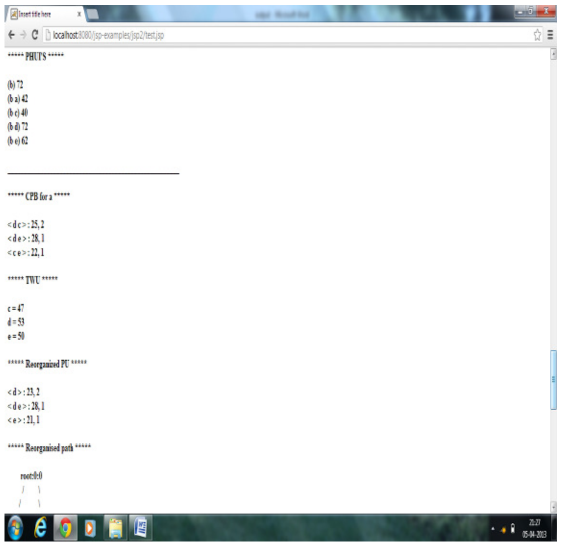
```
::: drawTransitionTree :::
***************************************************

amul = a
britannia = b
cadbory = c
Dairymilk = d
surfexcel = e
rin = f
power = g
dabor = h
          root:0:0
          /     \
         /       \
        /         \
       /           \
     c:11:2       e:18:4
      /           /    \
     /           /      \
   d:15:2      d:18:1  c:35:3
    /           /      /    \
  a:25:2      a:28:1  d:22:1  a:22:1
   /           /      /
  b:10:1      b:32:1  b:30:1

***************************************************
```

Figure A1.7: Result after construction of UP-Tree



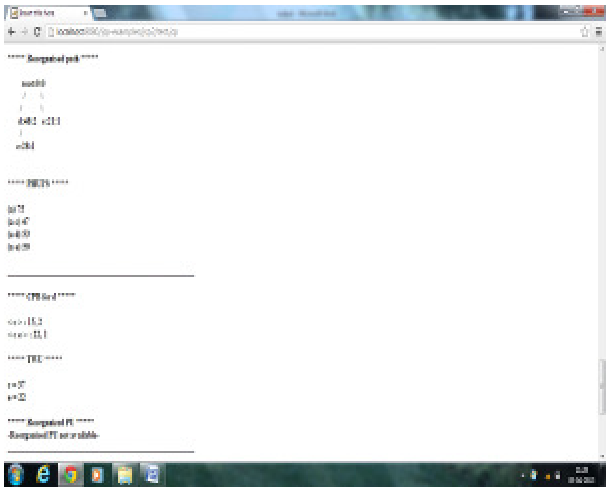
```
***** CPB for b *****

<a d c> : 10, 1
<a d e> : 32, 1
<d c e> : 30, 1

***** TWU *****

a = 42
c = 51
d = 72
e = 73

***** Reorganized PU *****

<d c> : 5, 1
<d e> : 27, 1
<d c e> : 30, 1

***** Reorganised path *****

     root:0:0
      /    \
     /      \
   d:4:1   e:52:2
    /        \
   /          \
  c:5:1     d:56:2
             /
          c:30:1
```

Figure A1.8: Result after construction of CPB



```
***** PHUI'S *****

(b) 72
(b a) 42
(b c) 40
(b d) 72
(b e) 62

______________________________

***** CPB for a *****

<d c>:25,2
<d e>:28,1
<c e>:22,1

***** TWU *****

c=47
d=53
e=50

***** Reorganized PU *****

<d>:23,2
<d e>:28,1
<e>:21,1

***** Reorganised path *****

    root:0:0
     /   \
    /     \
```

Figure A1.9: Result after discovery of PHUIs



```
***** Reorganised path *****

    root:0:0
     /   \
    /     \
  d:48:2  e:21:1
   |
  e:28:1

***** PHUI'S *****

(a) 75
(a c) 47
(a d) 53
(a e) 59

______________________________

***** CPB for d *****

<c>:15,2
<c e>:[illegible]

***** TWU *****

c=37
e=22

***** Reorganized PU *****
-Reorganized PT not available-
______________________________
```

Figure A1.10: Result after discovery of PHUIs

REFERENCES

[1] Ahmed.C. F, Tanbeer.S. K,Jeong.B.S and Lee.Y.K 'Efficient tree structures for high utility pattern mining in incremental databases',IEEE Transactions onKnowledge and Data Engineering, Vol. **21**, Issue **12**, pp. 1708-1721, **2009**

[2] Chan.R, Yang.Q and Shen.Y 'Mining high utility itemsets', in Proc. of Third IEEE Int'l Conf. on Data Mining, pp. **19-26, 2003**

[3] Cai.C. H, Fu.A. W. C, Cheng.C. H and Kwong.W. W 'Mining Association Rules with Weighted Items', in Proc. of the Int'l Database Engineering and Applications Symposium (IDEAS 1998), pp. **68-77**, **1998**

[4] Erwin.A, Gopalan.R. P and Achuthan.N. R 'Efficient mining of high utility itemsets from large datasets', in Proc. of PAKDD , LNA 5012, pp. **554- 561, 2008**

[5] Han.J,Pei.J, Yin.Y 'Mining frequent patterns without candidate generation',inProc. of the ACM-SIGMOD Int'l Conf. on Management of Data, pp. **1-12, 2000**

[6] Liu.Y, Liao.W and Choudhary.A 'A fast high utility itemsets mining algorithm', in Proc. of the Utility-Based Data Mining Workshop, **2005**

[7] Vincent S. Tseng, Bai-En Shie, Cheng-Wei Wu, and Philip S. Yu 'Efficient Algorithms for Mining High Utility Itemsets from Transactional Databases', IEEE Transactions on Knowledge and Data Engineering, **2012**

AUTHORS PROFILE

Mr.P.S.PRAKASH obtained his Bachelor's degree in Information Technology from "PGP College of Engineering and Technology - Namakkal" under Periyar University and Masters Degree in Computer Science and Engineering from "Kongu Engineering College of – Perundurai, Erode" under Anna University, Chennai. He has 7 years of teaching experience. Presently he is working as an Assistant Professor in the department of Information Technology in Kongu Engineering College, Perundurai. He has Published 1 and 2 Papers in International and National Conferences. His area of interest includes, Data Mining, Object Oriented Programming Concepts and Database Management System. He has organized workshops and seminar for the benefit of faculty members and students. He has attended 15 programmes like Seminars, FDP, Workshops and etc organized by various Engineering colleges.

Dr. S. ANANDAMURUGAN obtained his Bachelor's degree in Electrical and Electronics Engineering from "Maharaja Engineering College - Avinashi" under Bharathiyar University and Masters Degree in Computer Science and Engineering from "Arulmigu Kalasalingam College of Engineering – Krishnan Koil" under Madurai Kamaraj University. He completed his Ph.D in Wireless Sensor Networks under Anna University, Chennai. He has 13 years of teaching experience. Presently he is working as an Assistant Professor (Selection Grade) in the department of Information Technology in Kongu Engineering College, Perundurai. He is the life member of ISTE, CSI & ACEEE. He received "Best Staff" award for the year 2007-08. He has authored more than 70 books. He has Published 20 papers in International and National Journals and 10 Papers in International and National Conferences. His area of interest includes, Sensor Networks and Green Computing. He is an Editorial Board Member of International Journal of Computing Academic Research (IJCAR). He has organized 1CSIR sponsored seminar for the benefit of faculty members and students. He has attended 40 programmes like Seminars, FDP, Workshops and etc organized by various Engineering colleges.