**UG PROJECT REPORT**

**On**

***Efficient algorithms for mining closed and top-k High Utility Itemsets***

***Submitted by:***

**Gaurav Garg**

**Roll. No: 15045037**

**&**

**Sanjeet Giri**

**Roll. No: 15045088**

***Under the Supervision of:***

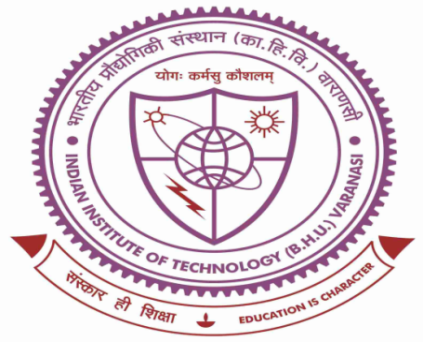
**Dr. Bhaskar Biswas & Dr. A.K. Verma**

***In partial fulfillment of the requirements for the degree of***

**Bachelor of Technology**

**In**

**Chemical Engineering**

****

**DEPARTMENT OF CHEMICAL ENGINEERING & TECHNOLOGY**

**INDIAN INSTITUTE OF TECHNOLOGY**

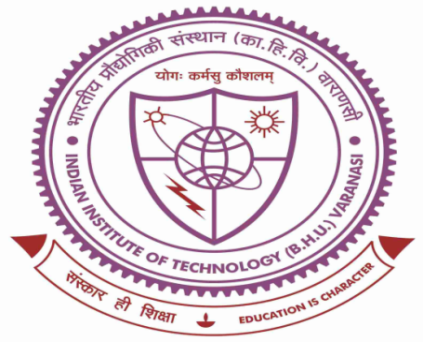
**(BANARAS HINDU UNIVERSITY)**

**VARANASI-221005 INDIA**

**21st April 2018**

***Contents***

|  |  |  |
| --- | --- | --- |
| **Chapter** | **Title** | **Page No.** |
|  | **Certificate** | **2** |
|  | **Acknowledgement** | **3** |
| **1** | **Introduction & Literature Review** | **4** |
| **2** | **Problem Statement** | **6** |
| **3** | **Algorithms** | **7** |
| **4** | **Experimental Results** | **9** |
| **5** | **Conclusion and Scope of Future Work** | **12** |
| **6** | **References** | **13** |

****

**DEPARTMENT OF CHEMICAL ENGINEERING & TECHNOLOGY**

**INDIAN INSTITUTE OF TECHNOLOGY (BHU)**

**VARANASI – 221005 INDIA**

**CERTIFICATE**

This is to certify that the UG Progress Report on **“Efficient algorithms for mining closed and top-k high utility itemsets”,** submitted by **Gaurav Garg (Roll No: 15045037) & Sanjeet Giri (Roll No: 15045088)** in partial fulfillment of the requirements for the degree of **Bachelor of Technology in Chemical Engineering** has been carried out under my supervision during the Even Semester 2017-18.

**Dr. A.K.Verma**

**Supervisor**

**Date: 21st April 2018**

**Place – Varanasi**

**Dr.Bhaskar Biswas**

**Co-Supervisor**

***Acknowledgement***

On the very outset of this report, we would like to extend our sincere & heartfelt obligation towards all the personages who have helped us in this endeavor. Without their active guidance, help, cooperation & encouragement, we would not have made headway in the project. We are ineffably indebted to Dr. A.K.Verma & Dr. Bhaskar Biswas for conscientious guidance and encouragement to accomplish this assignment. We are extremely thankful and pay my gratitude to PhD. Member of Computer Science & Engineering department Mr. Kuldeep Singh for his valuable guidance and support on completion of this project. We extend our gratitude to Chemical Engineering and Technology Department IIT BHU for giving us this opportunity. We also acknowledge with a deep sense of reverence, our gratitude towards our parents and members of our family, who have always supported us morally as well as economically. At last but not least gratitude goes to all of our friends who directly or indirectly helped us to complete this project report. Any omission in this brief acknowledgement does not mean lack of gratitude.

***Gaurav Garg***

***&***

***Sanjeet Giri***

**Date: 21st April 2018**

**Place – Varanasi**

***Introduction and Literature Review***

High Utility Itemset (HUI) mining problem involves the use of item utilities to discover profitable itemsets from a transactional database. It considers both internal and external utilities of items to discover profitable itemsets from the database. The problem has received significant attention in the recent years due to its potential applicability in numerous business and scientific applications.

Most of the current works in the literature support only items with positive unit profits. However, in most real-life situations, there is a need to consider items with either positive and negative unit profits or margins. For example, supermarket firms like Walmart often runs hundreds or thousands of cross product promotional campaigns per month. The campaign often involves offering products at everyday low pricing (EDLP), discounted price (that might lead to negative margin) or free products (negative profit) or bundled offerings (mix of discounted and non- discounted products). The additional costs (or losses) incurred on individual items that are part of a promotion are insignificant, if the overall promotional campaign delivers profitable outcomes. In essence, a firm is interested in choosing the bundle of products (or itemsets) that maximize its overall profitability.

An important limitation of traditional HUIM algorithms is that they often produce a huge amount of high-utility itemsets. Hence, it can be very time-consuming for users to analyze the output of these algorithms. Moreover, this makes HUIM algorithms sufferer from long execution times and even fail to run due to huge memory consumption or lack of storage space. To address this issue, it was recently proposed to mine a concise and lossless representation of all HUIs named closed high-utility itemsets (CHUIs). The concept of CHUI extends the concept of closed patterns from FIM. A CHUI is a HUI having no proper supersets that are HUIs and appear in the same number of transactions. This latter representation is interesting since it is lossless (it allows deriving all HUIs). Furthermore, it is also meaningful for real applications since it only discovers the largest HUIs that are common to groups of customers. However, CHUI mining can be very computationally expensive.

To precisely control the output size and discover the itemsets with the highest utilities without setting the thresholds, another promising solution is to redefine the task of mining HUIs as mining top-k high utility itemsets (top-k HUIs). The idea is to let the users specify k, i.e., the number of desired itemsets, instead of specifying the minimum utility threshold. Setting k is more intuitive than setting the threshold because k represents the number of itemsets that the users want to find whereas choosing the threshold depends primarily on database characteristics, which are often unknown to users.

There are two traditional algorithms followed to mine Top-k HUIs which are named TKU (mining Top-K Utility itemsets) and TKO (mining Top-K utility itemsets in One phase). But, these algorithms can only mine items with positive utilities. However, in real life these algorithms cannot be directly applied because they do not consider negative utilities. Therefore, we are proposing an algorithm that can mine Top-k HUIs of items with positive and negative utilities.

Therefore, in this project, two efficient algorithms are presented where one in FHNClosed algorithm which effectively mines Closed High Utility Itemsets (CHUIs) and another one is top-k FHN which mines the top-k HUIs with highest amount of profits. Both the algorithms have different input parameters and therefore different applications in real world. The main challenge faced here is mining items with negative profits as well.

***Problem Statement***

1. Closed High Utility Itemsets Mining

Traditional CHUIs mining algorithms only works with positive utility. However, in real life applications, as stated above with examples, negative utilities occur widely. In order to target this issue, we are proposing an algorithm FHNClosed for mining CHUIs with negative utilities. The key adoptions and contributions of this project are as follows:

1. A vertical list structure called PNU-list is used which maintains all the information required for mining HUIs without performing multiple time-consuming database scans. This list structure is adopted from FHN algorithm.
2. It adopts a novel sequence closure checking scheme called Bi-Directional Extension, and prunes the search space more deeply compared to the previous algorithms by using the Backward Extension, Forward Extension and closure jumping pruning methods and the Scan-Skip optimization technique adopted from BIDE+ algorithm.
3. An extensive experimental study is carried on several real-life datasets. Results show that the proposed algorithm outperforms the state-of-the-art algorithms in terms of run- time, memory consumption and scalability.
4. Top-k High Utility Itemsets Mining

Using a parameter k instead of the minutil threshold is very desirable for many applications. For example, to analyze customer purchase behavior, top-k HUI mining serves as a promising solution for users who desire to know “What are the top-k sets of products (i.e., itemsets) that contribute the highest profits to the company?” and “How to efficiently find these itemsets without setting the minutil threshold?”. Although top-k HUI mining is essential to many applications, developing efficient algorithms for mining such patterns is not an easy task. It poses three major challenges as discussed below.

1. In traditional HUI mining, the search space can be efficiently pruned by the algorithms by using a given minutil threshold. However, in the scenario of top-k HUI mining, no minutil threshold is provided in advance. Therefore, the minimum utility threshold is initially set to 0 and the designed algorithm has to gradually raise the threshold to prune the search space.
2. How to effectively raise the minutil Border threshold without missing any top-k HUIs? A good algorithm is one that can effectively raise the threshold during the mining process. However, if an incorrect method for raising the threshold is used, it may result in some top-k HUIs being pruned. Thus, minutil rising technique of TKO algorithm is used.
3. Lastly, to mine itemsets with negative utilities, PNU list data structure of FHN algorithm is used.

***Algorithms***

The main procedure of the proposed FHN algorithm takes as inputs (1) a transaction database D with quantity values, (2) the user-specified minUtil threshold, and (3) the user-specified profit-table ptable. The algorithm first scans the database to calculate the redefined TWU of each item (Line 1). The designed algorithm identifies the set I of all items having a TWU no less than minUtil. Other items are ignored since they cannot be part of a high utility itemset by Property 3 (Line 2). The TWU values of items are used to establish the total order on items from Definition 10, which is different from the TWU-ascending order suggested in (Line 3). A second database scan is then performed (Line 4). During this database scan, items in transactions are reordered according to the total order, the PNU-list of each 1-item I is built as well as the EUCS (Estimated Utility Co-Occurrence Structure). The original EUCS stores the TWU of all pairs of items {a, b} such that u ({a, b}) = 0. But the TWU is no longer an upper- bound on the utility of itemsets, when negative items are considered. To solve this issue in FHN, as previously explained, the EUCSis adapted to use the redefined TWU. Thus, items not in I ∗are dis- carded from the EUCS, and the EUCS can provide an upper bound on the utility of itemsets. In addition, as suggested in FHM, the EUCS is implemented as a hashmap of hashmaps since in practice a limited number of pairs of items co-occur in transactions.

The Search procedure takes as inputs (1) an itemset P, (2) extensions of P having the form Pz meaning that Pz was previously obtained by appending an item z to P, (3) minUtil and (4) the EUCS. The search procedure operates as follows. For each extension Px of P, it can be a CHUI only if it has no backward extension and no forward extension, this is checked as shown in the pseudo code. If the extension is an HUI and has no backward extension, it is sent as an output depending on closure jumping pruning strategy. After this, LA-prune is applied to check if any of its extensions belong to the HUI group. Note that if closure jumping is applicable this step is skipped because all the extensions are already included in CHUI, this is the application of closure jumping. Below is the pseudo code for Search Procedure of FHN algorithm.

**Algorithm 1: Proposed FHN Closed algorithm**

input: D : a transaction database; minUtil: a user-specified minimum utility threshold; ptable : a user-specified profit-table

output: The set of high-utility itemsets (HUIs)

Scan D to calculate the TWU of single items with the redefined RT U;

I ∗ ← each item i such that T W U(i ) ≥ minUtil;

Let \_ be the designed order on I ∗;

Scan D to build the PNU-list for each item i ∈ I ∗ and build the EUCS structure;

Search ( ∅ , I ∗, minUtil, EUCS) ;

return HUIs;

**Algorithm 2: The search Procedure - FHNClosed**

-------------------------------------------------------------------------------------------------------------------------------------

input: P:an itemset, ExtensionsOfP:set of extensions of P, the minUtil threshold, the EUCS structure

output: The set of CHUIs

foreach itemset Px in ExtensionsOfP do

to check backward extension

foreach extension Py < Px

if all tids of Px also lie in Py -> sup (Px)= sup(Pxy)

hasBackwardExtension = true;

to check forward extension and closure Jumping

foreach extension Py>Px

if all tids of Px also lie in Py -> sup (Px) = sup(Pxy)

hasForwardExtension = true;

if for some Py hasForwardExtension = false

closureJumping = false;

if Px.SUMIutils+ Px.SUMINutils >= minUtil && hasBackwardExtension do

if closureJumping do

output Px+(all Py such that Py>Px);

else if hasForwardExtension

ouput Px;

if Px.SUMputils+ Px.rutils>=minUtil &&!closureJumping then

ExtensionsOfPx ← ∅ ;

foreach itemset P y ∈ ExtensionsOfP such that y \_ x do

if ∃ T W U({ x, y } ) ∈ EUCS ∧ T W U({ x, y } ) ≥ minUtil then

Pxy ← P x ∪ P y ;

Pxy.UL ← Construct (P, P x, P y );

ExtensionsOfPx ← ExtensionsOfPx ∪ P xy

Search ( P x , ExtensionsOfPx , minUtil, EUCS);

end;

***Experimental results***

1. **FHN vs. FHN Closed**

Experiment was done with 2 datasets (chess\_negative and Mushroom\_negative) taken from SPMF website. The results show that the number of HUIs in FHNClosed is greatly reduced as expected while the candidate count remains the same, i.e. for the same number of itemsets, number of HUIs in output is very less in case of FHNClosed algorithm.

Below are the experimental observations:

The following graphs show that FHNClosed outperforms FHN in terms of time taken and number of HUIs generated.

In terms of Time taken, FHNClosed is approximately 2 times faster than FHN and number of HUIs generated is on an average 23 times less in case of FHNClosed.

1. **Mushroom\_negative**
2. **Chess\_negative**
3. **Top-k**

Since this is the only algorithm present which mines top-k HUIs with negative utility values, it cannot be compared with any other state-of-the-art algorithm. Therefore, below is the representation of Run-time by FHN top-k algorithm with mushroom\_negative itemsets.

***Conclusion and Scope of Future Work***

**Conclusion**

In this paper, we have studied the problem of mining closed and top-k high utility itemsets from transactional databases with negative unit profits. We have adopted PNU list structure from FHN algorithm and used Bi-directional extension method to mine closed itemsets among the HUIs. We have also adopted minimum utility threshold raising technique from TKO algorithm.

We have also performed an extensive experimental study on real-life datasets to compare the performance of FHNClosed with FHN and found out that FHNClosed outperforms in terms of run-time and number of HUIs.

**Scope of Future Work**

* These algorithms mine High Utility Itemsets, same concepts can be applied to Mining High Utility sequential patterns.
* Pruning strategies used in GHUI algorithm (A-prune & U-prune) can further be applied which reduces the run-time.
* PNU list structure is used here which is vertical data structure, using horizontal data structure i.e. Utility tree, may further reduce the run-time.

***References***

[1] P. Fournier-Viger, FHN: efficient mining of high-utility itemsets with negative unit profits, in: Proc. 10th Int. Conf. Advanced Data Mining and Application, 2014, pp. 16–29.

[2] Philippe Fournier-Viger1, Souleymane Zida2, EFIM-Closed: Fast and Memory efficient discovery of Closed High-Utility Itemsets.

[3] Wang, J., Han, J., Li, C.: Frequent closed sequence mining without candidate maintenance. IEEE Trans. on Knowledge Data Engineering 19(8), 10421056 (2007)

[4] Jianyong Wang\_and Jiawei Han, BIDE: Efficient Mining of Frequent Closed Sequences.

[5] Vincent S. Tseng, Senior Member, “Efficient Algorithms for Mining Top-K High Utility Itemsets”.

[5] M. Liu, J. Qu, Mining high utility itemsets without candidate generation, in: Proc. CIKM12, 2012, pp. 55–64 .

[6] S. Krishnamurthy, Pruning strategies for mining high utility itemsets, in: Ex- pert Systems with Applications, vol. 42, 2015, pp. 2371–2381.

[7] Zidaet. al. 2016 “EFIM: A Fast and Memory Efficient Algorithm for High-Utility Itemset Mining”.

[8] Yin et. al. 2012 “USpan: An Efficient Algorithm for Mining High Utility Sequential Patterns”.

[9] Liu et.al. 2012 “Mining High Utility Itemsets without Candidate Generation”.

[10] Zidaet. al. 2015 “Efficient Mining of High Utility Sequential Rules”.