An artificial bee colony algorithm to mine high utility item sets

Viveka S

Assistant Professor,
Department of Information Technology,
Velalar College of Engineering and Technology,
shiveka@gmail.com

Kalaavathi, B

Professor & HOD,
Department of Computer Science,
KSR Institute for Engineering and Technology,
kalabhuvanesh@yahoo.com

Abstract: Data mining tries to discover the interesting and useful patterns that are hidden it the huge databases. Periodic High utility itemset Mining is an emerging technique in the data mining which finds the high utility or high profitable items in the large databases based on time periods. The importance of the periodic high utility itemset mining is that it considers the factors profit, frequency and time period of occurrence of the itemset. The exponential search space is the crucial challenge in the High Utility itemset mining. In this paper an optimized algorithm Artificial Bee Colony (ABC) algorithm is used to mine periodic high utility itemset with fast convergence. The behavior of the honey bee is used as optimization technique. The proposed HUIM-ABC uses, Utility, Transaction Utility (TU)and Transaction Weighted Utilization(TWU) as the initialization factors for finding periodic high utility itemsets. From the experimental results it is seen than the proposed algorithm achieves better efficiency.

Keywords: Artificial Bee Colony, Periodic High Utility Itemset Mining.

1 Introduction

Knowledge discovery in database (KDD) is one of the major research areas. It is because that the data contains many hidden and highly useful information. Many organizations have huge data, and these data can be analyzed to extract the useful information which is hidden inside. These data exponentially increases day by day. Extracting of the data can be done using different kinds of data mining techniques such as Association Rule Mining (ARM), Clustering, Classification, Linear Regression etc.,. The Frequent Pattern Mining (FPM) is a part of data mining which finds the frequently occurring patterns in the databases. ARM finds the associations that exist between the frequent patterns. Both the frequent pattern and ARM finds the information based on the threshold value given as the user input. Out of the many data mining algorithms ARM attracted many researchers and number of algorithms has been developed in ARM. ARM considers only the frequency of the item but does not consider the utility of the item. In marketing, web stream databases the profit of the items also plays an important role. An item may be infrequent in a database but it may generate a high profit. These infrequent high profit item needs to be considered in finding the itemset. High Utility Itemset Mining (HUIM) is a successor of ARM which finds the high profitable items. Instead of finding the frequent pattern above the user specified threshold HUIM finds the High Utility Itemsets(HUIs). An itemset is said to High Utility Itemset (HUI) if the profit of the item in the database is greater than or equal to the user specified threshold profit. The utility measurement can be in terms of utilitzation, cost, profit and others. The major challenge faced in HUIM is that the downward closure property is not satisfied. In FPM and ARM in finding patterns during the first step, the 1-candidate itemset are found, and the i+1-candidate patterns are found from the i-candidate patterns using the downward closure property. The search space of i+1-candidate itemset is limited to i-candidate itemset but this property is not satisfied in the HUIM. The search space of finding HUIM grows exponentially with the length of the patterns. Many algorithms have been introduced to find the HUIM efficiently. The two phase miner is a state of art algorithm which mines the HUIM effectively in two phases. Downward closure property using transaction weight has been used in this two phase miner to effectively prune the search space. An improved version of the algorithm FHM uses a structure EUCS based on co-occurrence pruning to reduce the search space. Tree based algorithm such as UP tree and UPtree + have been proposed by reducing the search space by upper bounds. Another tree based algorithm is proposed by Lin et al. which used condensed high-utility pattern (HUP)-tree. FP tree and TWU based algorithm for mining HUI is proposed by (Han et al. 2004). (Lan et al. 2013) proposed a indexed projection of the dataset for mining HUI, this algorithm used a different approach of projecting pseudo database for every iteration instead of depth first approach used in tree based algorithms. These algorithms does not considers the period of occurrence

of an itemset and these algorithms cannot be directly applied to PHUI mining. To address these issues periodic high utility mining is proposed. In this paper a new algorithm PHUIM-ABC is proposed to mine all the high profit items that occurred in the transaction during the specified period. Organization of paper is as follows: Section 2 describes the related work, section 3 describes the PHUIM-ABC algorithm, Section 4 describes the experimental study and section 5 gives the conclusion.

2 Related work

The basic terms used in the HUIM are defined below using the standard conventions as in the previous work (Ahmed et al., 2009; Liu & Qu, 2012; Liu et al., 2005; Yao & Hamilton, 2006).

Let $C=\{c_1, c_2, c_3, \ldots c_n\}$ be set of distinct commodities or items in the dataset D. A transaction in a dataset can be defined as set of commodities that belongs to C given by $T_i=\{c_n \mid n=1,2,3\ldots N, c_k \in C\}$ where N is the number of commodities that participate in the T_i transaction. A dataset D is defined as the set of transaction that are present $D=\{T_1, T_2, \ldots T_m\}$ where the total number of transactions in dataset D is given by m.

Let D be a small transaction dataset and the transaction history of D is given in Table 1.

Definition 1. For each item $c_i \in C$ is associated with a profit value / external utility value referred as $P(c_i)$. From Table 1, the $P(A)=5$

Definition 2. For each item $c_i \in C$ is associated with a purchase quantity / internal utility value referred as $I(c_i,T_i)$. From Table 1, the $I(A,T_1)=1$

Definition 3. Utility/Value of an item in a transaction.

The utility of an item c_i in a transaction is identified by using two factors profit and quantity. The utility of an item is denoted by $V(c_i)$

$$V(c_i,T_{id}) = P(c_i) * Q(c_i,T_{id}) \qquad (1)$$

For the example in table 1 the utility of item B in transaction T_3 is calculated as

$$V(B, T_3) = 10$$

Table 1 Transaction Dataset D

T_{id}	Transaction	Profit / Utility (P)	Transaction Utility (TU)
T_1	A1,C1	5,1	6
T_2	E1	3	3
T_3	A1,B5,C1,D3,E1	5,10,1,6,3	25
T_4	B4,C3,D3,E1	8,3,6,3	20
T_5	A1,C1,D1	5,1,2	8
T_6	A2,C6,E2	10,6,6	22
T_7	B2,C2,E1	4,2,3	9

Definition 4. Utility/Value of an itemset in a transaction.

The utility of an itemset co_k in a transaction is calculate as sum of the utility of each item in the itemset in a transaction, provided if all the item in the itemset participates in the transaction . The utility of an itemset is denoted by $V(co_k, T_{id})$

$$V(co_k, T_{id}) = \sum_{c_i \in co_k \wedge co_k \subseteq T_{id}} V(c_i, T_{id}) \qquad (2)$$

For the example in table 1 the utility of item B in transaction T_3 is calculated as

$$V(B, T_3) = 10$$

Definition 5. Utility/Value of an itemset in a dataset D.

The utility of an itemset co_k in a dataset is calculate as sum of the utility of each item in the itemset in all the transaction in the dataset, provided if all the item in the itemset participates in the transaction . The utility of an itemset is denoted by $V(co_k)$

$$V(co_k) = \sum_{co_k \subseteq T_{id} \wedge T_{id} \subseteq D} V((co_k, T_{id}) \qquad (2)$$

For the example in table 1 the utility of item B in transaction D is calculated as

$$V(B) = 22$$

Definition 6. High Transaction Weighed Utility/Value of an itemset in a dataset D.

An itemset co_k is said to be a HTWUI if the sum of the transaction utility of itemset in all the transaction is greater than or equal to the user specified threshold (Mthreshold). The HTWUI can be represented as follows

$$\text{HTWUI}((co_k) \leftarrow \{(co_k | \sum_{(co_k \subseteq T_{id} \wedge T_{id} \in D} TU(co_k, T_{id}) \geq Mthreshold\} \quad (3)$$

For the given example in table 1, let the user specified minimum threshold mthreshold = 24. For the itemset AC in D, the utility of itemset AC is calculated as

$$TU(AC,D) = TU(T_1) + TU(T_3) + TU(T_5) + TU(T_6)$$

$$TU(AC,D) = 6 + 25 + 8 + 22 = 61$$

$TU(AC,D) \geq$ Mthreshold i.e., $61 \geq 25$, Hence the itemset AC in D is a HTWUI.

Definition 7. High Utility Itemset (HUI) in a Dataset D

An itemset co_k is said to be a HUI if the sum of the utility of itemset in all the transaction is greater than or equal to the user specified threshold(Mthreshold). The HUI can be represented as follows

$$\text{HUI}((co_k) \leftarrow \{(co_k | \sum_{(co_k \subseteq T_{id} \wedge T_{id} \in D} V((co_k, T_{id}) \geq mthreshold\} \quad (3)$$

For the given example in table 1, let the user specified minimum threshold mthreshold = 24. For the itemset AC in D, the utility of itemset AC is calculated as

$$V(AC,D) = V(AC,T_1) + V(AC,T_3) + V(AC,T_5) + V(AC,T_6)$$

$$V(AC,D) = 6 + 15 + 6 + 16 = 43$$

V(AC,D) ≥ Mthreshold i.e., 43 ≥ 25, Hence the itemset AC in D is a HUI.

Definition 7. Transaction Weighted Utility (TWU) of an itemset in Dataset D

Transaction Weighted Utility (TWU) of an itemset co_k in Dataset D can be defined as the sum of the transaction utility of the itemset co_k in the all the transactions in D.

$$TWU(co_k) = \sum_{co_k \subseteq T_{id} \wedge T_{id} \in D} TU(T_{id}) \qquad (4)$$

$$TWU\ (AC) = 6+25+8+22 = 61$$

3 Proposed Phuim-ABC

Based on the behavior of honey for collecting the nectar, Artificial Bee Colony algorithm is based on the honey bee swarm intelligence. In this model, the bees are classified into three categories, onlooker bees, employed bees and scout. Employed bees are the one which goes to the food position and the information regarding the food position and the amount of nectar is shared with the onlooker bee. Onlooker bee waits in the dancing area of the hive and decides the food source based on the amount of the food available. Scout bee carries the random search to find the food sources.

3.1 Preprocessing Phase

In the designed HUIM-ABC algorithm, the TWU model (Liu et al. 2005) of traditional HUIM is first adopted to discover high-transaction-weighted utilization 1-itemsets (1-HTWUIs). Based on the transaction-weighted downward closure (TWDC) property of HTWUIs, the unpromising items can be efficiently pruned. Thus, the computations for discovering HUIs can be greatly improved. The phase to discover 1-HTWUIs is described below. First, the utility of items of each transaction is calculated as the transaction utility (*tu*).

The transaction-weighted utility of an item is thus calculated by summing up the transaction utility if an item appearing in the transaction. This process is used to estimate the upper bound value of an item. If the transaction-weighted utility of an item is no less than the minimum utility count, it is considered as a HTWUI.

PHUIM Set

The itemset of the PHUIM is associated with the min Period, maxPeriod, avgPeriod, fitness value.

Fitness function

Fitness function utility value of each itemset.

Fitness (X)=U(c) which is same as defined in the definition 3

The fitness value of the itemset is given as the nector amount of the employed bee

3.2 Evaluation Phase

Evaluation of Fitness Function

The onlooker bee evaluates the fitness function in order to decide the food source, the decidability is done based on the probability function given by the equation below

$$\text{Prob}_x = \frac{f(X)}{\sum_{n=1}^{SN} f(X)}$$

Where f(X) is the value of fitness or amount of nectar given by the employed bee and the number of food sources known to the bee is given by the SN.

PHUIM Discovery

The steps in the Artificial bee intelligence algorithm is given below:

Step 1: initialize the fitness function and find the transaction utility of all items present in D.

Step 2: Find the number of employed bees and put the found employed bees in the food source.

Step 3: Also place the onhooker bee on the food source in the memory

Step 4: Discover the new food sources by searching using the scout bees.

Step 5: Repeat from step 2 untill termination condition is met.

Algorithm: PHUIM-ABC

Input: *A transaction dataset D, Profit table P, Minimum Utility Threshold Mthreshold*

Output: *PHUIs*, a set of high-utility itemsets.

1 for *each* $T_i \in D$ **do**

2 for *each* $c_{ij} \subseteq T_q$ **do**

3 $tu(T_q) = I(C_{ij}, T_i) \times ptable(C_{ij})$;

calculate $twu(C_{ij}) = C_{ij} \subseteq T_q$ and $tu(T_q)$;

5 $TU = T_q\ q \in D\ tu(T_q)$;

6 find 1-HTWUIs$\leftarrow$ { C_{ij} |$twu(C_{ij}) \geq TU \times \delta$};

7 set k = |1-HTWUIs|
8 set number of employed bee to k;
9 for $i \leftarrow 1$ *to* M **do**
10 for $j \leftarrow 1$ *to* k **do**
initialize p_j
11 $I(t)$ = either 0 or 1
12 calculate the fitness value as f(i)
13 initial the velocity of particles in (0, 1)
14 if $fitness(pi\ (t)) \geq TU \times \delta$ **then**
15 $PHU\ I\ s \leftarrow Get\ \ Item(pi\ (t)) \cup PHUI\ s$
16 find a PHUI
17 find *best (employed bee)* of each M employed
18 while *termination criteria is not reached* **do**
19 for $i \leftarrow 1$, M **do**
20 update the $prob\ (t + 1)$ probabilities of employed bee
21 if $fitness(pi\ (t + 1)) \geq TU \times \delta$ **then**
22 $PHU\ I\ s \leftarrow GetItem(pi\ (t + 1)) \cup PHUIs$
23 find a PHUI
24 find *best (employed bee)* of each M employed
25 set $t \leftarrow t + 1$;
26 return *PHUIs*;

4 Experimental Results

All the algorithms are implemented in the Java API, in a computer system of 1 GB RAM. For the performance evaluation the experiment is conducted in the real life datasets, foodmart, mushroom and retail datasets.

The foodmart is a sparse dataset with around 1K items and 4K transactions. The mushroom is a dense dataset with around 100 items and 8K transactions. The retail is a sparse dataset with around 16K items and 88K transactions.

The performance of the proposed algorithm is measured in terms of execution time and memory utilization.

The proposed algorithm is compared with the state of art of algorithm HUI miner. The number of visited by using indexed structure PHUIM-ABC is more than 50 times lesser than HUI miner. The tighter upper bound reduces the number of candidate itemset reduced is around 20% considering the three datasets.

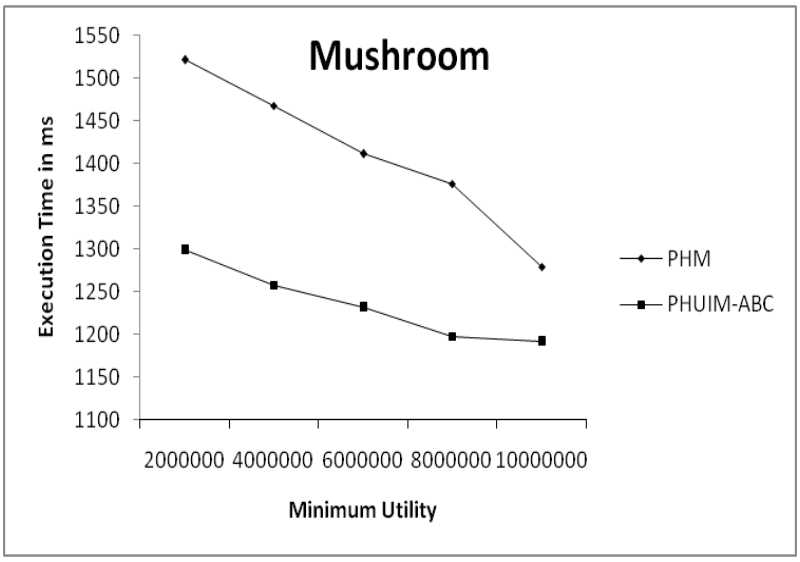

Figure 1 Execution Time of Mushroom Dataset

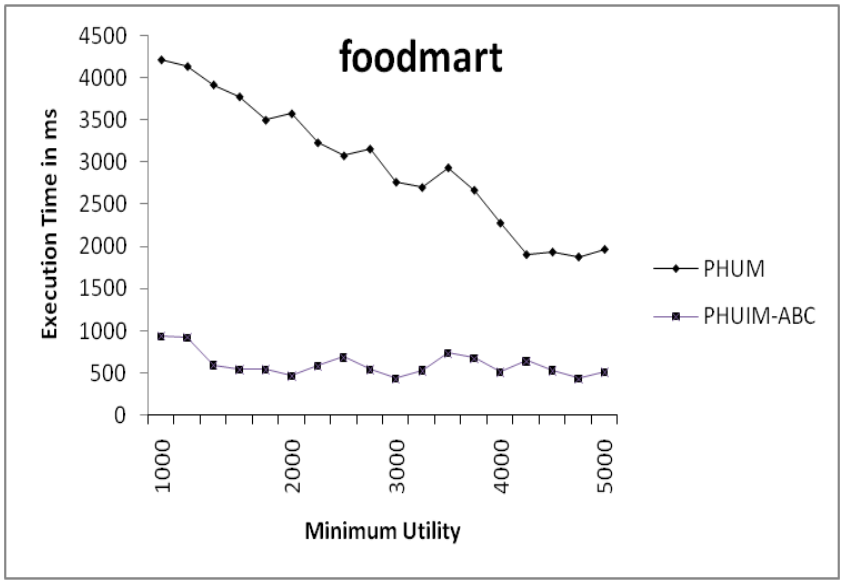

Figure 2 Execution Time of Foodmart Dataset

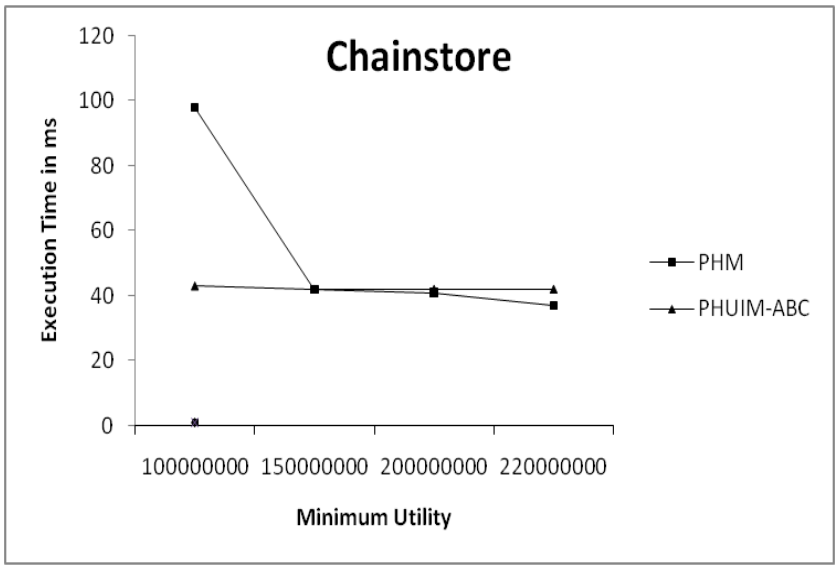

Figure 3 Execution Time of Chainstore Dataset

From the experimental results it is clearly seen that the PHUIM-ABC performs better than the HUI miner. In mushroom dataset from Figure 1 the PHUIM-ABC performs better in all utility ranges. The reason is that the number of candidate visited in the tree is much pruned in the dense dataset. In foodmart dataset from Figure 2 the execution time is much improved in the order of 5 times faster than HUI miner. The performance is improved in sparse dataset also. From Figure3 it can be noted that there is only a slight improved in the order of 10 % compared to HUI miner.

5 Conclusion

From the experimental results it clearly shows the ABC based algorithms performs better than the state of art PHUIM algorithms. In future it is planned to implement GA algorithms for PHUIM to improve efficiency. This algorithm can be also extended to sequential utility mining algorithms

References

Agrawal, R., Imielinski, T., & Swami A. 1993. Mining association rules between sets of items in large databases. Proceedings of Special Interest Group on Management of Data, 22 (2) (1993), 207-216.

Agrawal, R., & Srikant, R. 1994. Fast algorithms for mining association rules in large databases. In: Proc. Int. Conf. Very Large Databases, (pp. 487-499), 1994.

Ahmed, C., Tanbeer, F, S. K., & Jeong B. S. 2010. A novel approach for mining high-utility sequential patterns in sequence databases," Electronics and Telecommunications Research Institute journal, vol. 32(5) (2010), 676-686.

Amphawan, K., Lenca, P., & Surarerks, A. 2009. Mining top-k periodic-frequent pattern from transactional databases without support threshold, in: Advances in Information Technology, Springer, (2009), 18-29.

Amphawan, K., Surarerks, A., & Lenca, P. 2010. Mining periodic-frequent itemsets with approximate periodicity using interval transaction-ids list tree. In: Proc. 2010 Third intern. Conf. on Knowledge Discovery and Data Mining, (pp. 245-248), 2010.

Ansari, E., Dastghaibifard G.H., & keshatkaran M. 2008. Distributed Trie Frequent Itemset Mining. Proceedings of International Multi Conference of Engineers and Computer Scientists, 1 (pp. 978-988), 2008.

Ashrafi, M. Z., Taniar D., & Smith K. 2004. Optimized Distributed Association Rule Mining", Proceedings of IEEE distributed system online, 5 (3) (2004), 1-18.

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), (2003).

Christian B. 2005. An Implementation of the FP-Growth Algorithm. In Proceedings of the 1st International Workshop on Open Source Data Mining: Frequent Pattern Mining Implementations, (2005), 1-5.

David, W. C., Jiawei Han, Vincent T. Ng, Ada W. Fu, & Yongjian Fu. 1996. A Fast Distribution Algorithm for Mining Association Rule. In Proceedings of Parallel and Distributed Information Systems, (1996), 31-42.

Fournier-Viger, P., Wu, C.W., Zida, S., & Tseng, V. S. 2014. FHM: Faster high-utility itemset mining using estimated utility co-occurrence pruning. In: Proc. 21st Intern. Symp. on Methodologies for Intell. Syst., (2014), 83 – 92.

Han Jiawei, Guozhu Dong & Yiwen Yin, 1999. Efficient mining of partial periodic patterns in time series database. Proceedings 15th International Conference on Data Engineering (Cat. No.99CB36337), Sydney, NSW, (pp. 106-115), (1999).

Indyk, P., Koudas N., & Muthukrishnan S. 2000. Identifying Representative Trends in Massive Time Series Data Sets Using Sketches. Proc. 26th Int'l Conf. Very Large Data Bases, (2000).

Kiran, R. U., & Reddy, P. K. 2009. Mining Rare Periodic-Frequent Patterns Using Multiple Minimum Supports. In: Proc. 15th Intern. Conf. on Management of Data 5 (6) (2009), 7-8.

Lan, G. C., Hong, T. P., & Tseng, V. S. 2014. An efficient projection-based indexing approach for mining high utility itemsets. Knowledge and Information Systems, 38 (2014), 85–107.

Liu, M., & Qu, J. 2012. Mining high utility itemsets without candidate generation. In: Proc. 22nd ACM Intern. Conf. Info. and Know. Management, (pp. 55-64), 2012.

Liu, Y., Liao, W., & Choudhary, A. 2005. A two-phase algorithm for fast discovery of high utility itemsets. In: Proc. 9th Pacific-Asia Conf. on Knowl. Discovery and Data Mining, (pp. 689-695), 2005.

Rakesh Agrawal, Christos Faloutsos & Arun N. Swami. 1993. Efficient Similarity Search In Sequence Databases. FODO '93 Proceedings of the 4th International Conference on Foundations of Data Organization and Algorithms, Springer, (pp. 69-84), 1993.

Song, W., Liu, Y., & Li, J. 2014. Fast and memory efficient mining of high utility itemsets based on bitmap. Intern. Journal of Data Warehousing and Mining, 10(1) (2014), 1-15.

Surana, A., Kiran, R. U., & Reddy, P. K. 2012. An efficient approach to mine periodic- frequent patterns in transactional databases. In: Proc. 2011 Quality

Issues, Measures of Interestingness and Evaluation of Data Mining Models Workshop, (2012), 254 – 266.

Tanbeer, S. K., Ahmed C. F., Jeong B. S., & Lee Y. K. (2009), Discovering periodic-frequent patterns in transactional databases, in Advances in Knowledge Discovery and Data Mining, Springer, (2009), 242-253.

Uday, U. R., Kitsuregawa, M., & Reddy, P. K. Efficient Discovery of Periodic-Frequent Patterns in Very Large Databases. Journal of Systems and Software, 112 (2015), 110-121.

Viger, P. Fourier, Lin J.CW., Duong QH., & Dam TL. 2016. PHM: Mining Periodic High-Utility Itemsets. In: Perner P. (eds) Advances in Data Mining. Applications and Theoretical Aspects. ICDM 2016. Lecture Notes in Computer Science, Springer, Cham (2016).

Yongwei Ding, Xiaohu Yang, Kavs, A.J., & Juefeng L. 2010. A novel piecewise linear segmentation for time series', Proceedings of the Second International Conference on Computer and Automation Engineering 4 (pp.52-55), 2010.

Zida, S., Fournier-Viger, P., Lin, J.CW., Wu, C.W., Tseng, V.S. 2015, EFIM: A Highly Efficient Algorithm for High-Utility Itemset Mining. In: Sidorov G., Galicia-Haro S. (eds) Advances in Artificial Intelligence and Soft Computing. Lecture Notes in Computer Science, (2015).