Received: 15 June 2025 | Revised: 23 June 2025 | Accepted: 25 June 2025 | Vol. 1 | Issue 1 | June 2025
e-ISSN: 3105-2169

International Journal of Mechatronics, Robotics, and Artificial Intelligence
Original article

Mining Dynamic Profit Databases: Efficient Solutions for High Utility Periodic and Closed Patterns

Arkan A. Ghaib[1], **Abdullah A. Nahi**[1], **Hussain A. Younis**[2,3 *], **Takaaki Fujita**[4]
[1] Department of Information Technologies Management, Southern Technical University, Basrah, Iraq
[2] College of Education for Women, University of Basrah, Basrah 61004, Iraq
[3] School of Computer Sciences, Universiti Sains Malaysia, Gelugor 11800, Malaysia
[4] Independent Researcher, Shinjuku, Shinjuku-ku, Tokyo, Japan

Correspondence
*Hussain A. Younis
College of Education for Women, University of Basrah
Email: hussain.younis@uobasrah.edu.iq

Abstract
High utility pattern mining (HUPM) is one of the key areas in data mining, which is concerned with identifying patterns with high utility from transactional databases. The temporal factors such as periodicity and recency along with dynamic variations in profits have recently been added to pattern mining. However, no methods so far unify these dimensions in a common framework. To this end, in this paper we propose the DTU-Miner algorithm that integrates temporal constraints and dynamic profit updates to overcome such limitations. Through the use of advanced data structures such as UPR-List and P-set and the introduction of some novel pruning strategies, DTU-Miner surpasses state of the art in terms of Runtime, Memory and pattern quality. Results on benchmark datasets show that DTU-Miner outperforms state-of-the-art algorithms, CPR-Miner and iEFIM-Closed, which suggests the effectiveness of DTU-Miner over dense and sparse datasets including dynamic attributes.

Keywords
High utility itemset, Pattern mining, Closed pattern, Closed itemset, Dynamic profit

I. Introduction

Frequent pattern mining (FPM) is one of the basic research directions in data mining, which aims to mine repeated patterns from transaction databases [1]. Despite having a widespread applicability, traditional FPM methods cannot satisfy the real world needs since they assume that all items are equally important and do not consider how many items are there in each transaction. These phenomena prompted the birth of high utility pattern mining (HUPM) [2][3] which considers item quantities and utility values such as profit to discover the patterns with higher relevance in practice.

Although HUPM has addressed the limitations of FPM to a great extent, it has also given rise to new challenges. It is worth mentioning that HUPM does not preserve the downward closure, which means that a subset or a superset of a high utility pattern may not be high utility patterns. Such an approach frequently produces a massive quantity of candidate patterns, which in turn incurs a high computational cost. Many algorithms have been proposed to alleviate this issue, including those using transaction-weighted utility (TWU) and utility-list structures to prune the search space [4]. Arkan et al. 2016 proposed the EFP-growth and MRFP-growth algorithms to solve the problem of discovering frequent patterns with massive datasets. The set of this kind of pattern is a concise representation of all PFPs [5][6].

Researchers have recently proposed different ways of improving the practicality of HUPM by incorporating new constraints like temporal and dynamic profits. The role of temporal considerations such as periodicity and recency has been especially important in domains such as system update over time. For example, CPR-Miner implements temporal constraints to mine closed periodic recent high utility patterns (CPRHUPs) which result in short and significant information for decision-makers within time which is very crucial in many extreme conditions [7] The same goes for the approach in iEFIM-Closed, which deals with the dynamic profits problem where utility values vary depending on external conditions such as promotions or changes in the supply chain, using condensed data structures and high

performance pruning techniques[8], improving the efficiency of Distributed Utility itemsets mining in relation to big data (IDUIM) Enhancing N-list, an efficient algorithm for mining frequent item collections; it encodes collections by means of an N-list and similarly identifies the frequently recurring collections directly by means of a set-enumeration search tree. Table I. is an instance of a transactional database where each row corresponds to a transaction (TID) and contains items along with their quantity purchased. For instance, example T1 includes items b, c, d, g with quantity of {1; 2; 1; 1} respectively. This is a static dataset with unit profits set to the same level for all items. Table II shows a transactional database with dynamic unit profits, where the value of the profit of different transactional items varies across transactions. For example, in T5, item b has a unit profit of $2.4, while in T1, it has a profit of $2. As profit values continuously vary, it leads to distinctive challenges in the computation of utility in terms of algorithm being dynamic towards such changing profit values. profit-based considerations. The timeline data are not only temporally insensitive, but they also impede extracting patterns that matter over time and economically. In order to fill this gap, we propose the Dynamic Temporal Utility Miner (DTU-Miner) algorithm. DTU-Miner, which integrates temporal constraints with dynamic profit modifications and utilizes sophisticated data structures to address high utility patterns mining in dynamic and time-critical data sets, is proposed as a consolidated approach for mining such patterns.

TABLE I.
A TRANSACTIONAL DATABASE (STATIC PROFITS)

TID	Transactions (Items and Quantities)
T1	{b: 1, c: 2, d: 1, g: 1}
T2	{a: 4, b: 1, c: 3, d: 1, e: 1}
T3	{a: 4, c: 2, d: 1}
T4	{a: 5, b: 2, d: 1, e: 2}
T5	{a: 3, b: 4, c: 1, f: 2}

TABLE II.
TRANSACTIONAL DATABASE CONTAINS DYNAMIC UNIT PROFITS

TID	Transactions (Items and Quantities)	Unit Profits (Per Item)
T1	{b: 1, c: 2, d: 1, g: 1}	{b: 2, c: 1, d: 5, g: 1}
T2	{a: 4, b: 1, c: 3, d: 1, e: 1}	{a: 1, b: 1.9, c: 0.9, d: 4.8, e: 4}
T3	{a: 4, c: 2, d: 1}	{a: 1.1, c: 1.1, d: 5.5}
T4	{a: 5, b: 2, d: 1, e: 2}	{a: 1.1, b: 2.2, d: 5.5, e: 4.4}
T5	{a: 3, b: 4, c: 1, f: 2}	{a: 1.2, b: 2.4, c: 1.2, f: 3.6}

II. RELATED WORK

A. *Preliminaries: Mining High Utility Patterns*

HUPM was proposed for frequent pattern mining on interesting itemsets given its roots in utility-based pattern mining, and research in the area has evolved since then. Two-Phase and HUI-Miner [9] are some early algorithms which were based on Transaction Weighted Utility (TWU) [10] to prune search space. However, these techniques often generate many candidate patterns with high computational cost.

Current Illustrative Efforts with HUPM Recent progress is attempted to be utilized on context-based applications. EFIM (Efficient High Utility Itemset Mining) (and its extension, EFIM-Closed) [11], introduces new pruning strategies, including Closure Jumping (CJU) and Efficient Utility Candidate Storage (EUCS), which aim to reduce the number of database scans. UP-Growth [12] and FHM [3] are some tree-based and list-based techniques that perform an optimization for utility mining. Dynamic Utility (HU) Items Dynamic Utility High-Utility Itemset Miner (DU-HUIM) algorithm which efficiently takes care of these dynamic utility variations.

B. *Mining Periodic Patterns*

Periodic pattern mining is important because it is used to discover trends that occur within specific periods of time and provides accurate and in-time information for temporal databases. One of the first periodic frequent pattern (PFP) [13] mining algorithms was PF-growth [12]; it considered some constraints such as MaxPer and MinSup [14]. But these restrictions often pruned too many of the nice patterns. Partial periodic pattern mining as well as more flexible models such as stable periodic-frequent patterns and maximum periodic patterns were proposed to solve this drawback. Since 5 years PHUP proposed to search for periodic high utility patterns in sequential data and introduced a novel criteria for with which occasional patterns and a comparatively increased the search of periodical with high utility patterns (after MinPer and AvgPer in 12) [12] for periodic mining [15].

C. *Frequent Constraints within HUPM*

HUPM uses several of these commonly applied constraints to produce more interpretable and efficient pattern mining results. Closed high utility patterns (CHUPs) [16] are a more concise representation of HUPs, as they will not allow for a proper superset of a CHUP to have the same support. RUP-Miner and similar methods [7] include recency constraints [10] in order to focus directly on those patterns that are persistent over time; they operate by preferentially expanding those patterns that have more recent transactions. Adding such constraints along with periodicity and dynamic profits can significantly expand the mined patterns to their useful forms in practice. Unified Search and Return Algorithm (USRA) that combines both capabilities into a single architecture. Through harnessing hybrid data structures and dynamic pruning techniques.

D. *Patterns Mining and Time Factor*

Trend dynamics in data is fueled by temporal signal such as periodicity & recency. Closed Recent High Utility Pattern Mining (CPRHUP) is the latest high utility pattern mining research, and it has led to the development of a novel mining algorithm in the form of CPR-Miner. To address this limitation, pruning methods such as TA-prune and

CUB-prune target dense datasets [17]. This approach has been very effective for applications where time is of the essence such as retail and customer behavior modelling [18].

E. Dynamic Profit Databases

HUPM has a distinctive challenge because, in dynamic profit databases the value of the utility changes with time, or in other words, the utility is tied to the moment it is generated, but the utility for each moment in the future can change due to external influences. To deal with this, iEFIM-Closed presents a new utility measure and an efficient database format to handle computation overhead, reducing it considerably [19]-[22]. It will enable the above mentioned utilities to be beneficial in real world applications, where the prices and costs change frequently.

III. PRELIMINARIES AND PROBLEM FORMULATION

This section presents basic definitions and also formally defines the problem of mining closed periodic recent high utility patterns (CPRHUPs). This work concentrates on the use of quantitative temporal databases in which all transactions are time-stamped and each item has one or more utility values. Such definitions include transaction utility (TU), actual utility (AU), and transaction-weighted utility (TWU), which are used as the basis for recognizing high utility patterns.

The mining process incorporates key constraints utility, periodicity, and recency. This definition sets limits for utilities (minUtil), periodicity (minPer, maxPer), and recency (minRe) to direct the search to worthwhile patterns. With these constraints in place, the CPRHUP mining framework seeks to achieve an optimal compromise between the expressiveness of patterns and the meet computational cost. New pruning approaches and utilizing data structures like UPR-Lists assist in reducing the mining process.

A. Definitions

1. TidSet: For a database DB and the pattern X, the transaction set TS(X) refers to the set of transactions containing X, defined as: $TS(X)=\{Tid(T)|T\in DB, X\subseteq T\}$, where $|TS(X)$ denoting the support count of X (called Sup(X)).
2. Actual Utility (AU): The actual utility of a pattern X in a transaction Tk can be represented as $AU(X,Tk)=\sum i\in XAU(i,Tk)$ The actual utility of X in the entire database is given by $AU(X)=\sum Tk\in TS(X)AU(X,Tk)$.
3. High Utility Pattern: Let's assume mini is the minimum utility threshold, then a pattern X is said to be a high utility pattern if $AU(X)\geq \mu\_ID$.
4. Transaction Weighted Utility (TWU): The TWU for a pattern X, is the summation of the transaction utility for the transactions where X is contained in the transaction, defined as $TWU(X)=\sum Tk\in TS(X)TU(Tk)$.
5. $\prec$-Extension: A relation for two items i and j forms a relation $i\prec j$ if TWU(i).
6. Generalized Extension (g-Extension): A pattern X' is a g-extension of X if $X\subset X'$, and every $j\in X'\backslash X$ satisfies $j\succ x1$, where x1 is the first item in X.
7. Remaining Utility (RU): Given a pattern X within a single transaction Tk , where items in Tk are arranged in order of itemset by some transactional ordering relation $\prec$, the remaining utility is defined as $RU(X,Tk)=\sum i\in Tk\backslash XAU(i,Tk)$ and $i\succ x$ for all $x\in X$.
8. Closed pattern: A pattern X is said to be closed if there does not exist any superset Y of X which is supported by the same transactions as X.
9. CPRHUP: A pattern P is called Closed Periodic Recency High Utility Pattern (CPRHUP) if it satisfies utility ($AU(P)\geq \mu$), periodicity ($minAvg\leq AvgPer(P)\leq maxAvg, MaxPer(P)\leq maxPer, MinPer(P)\geq minPer$) and recency ($Re(P)\geq \rho$) conditions.
10. Adaptive Utility: Utility values change over time, requiring immediate application of the appropriate profit utility.

B. Problem Statement

High Utility Pattern Mining (HUPM) is an important branch of data mining, which is a broad field that searches for patterns with high utility from the transaction database. Despite this criticality, methods based on traditional approaches suffer from the issues of excessive patterns generation, memory consumption, and high computation overhead, hindering their application in the real world. Moreover, these techniques work typically under static assumptions and do not take into account crucial dynamics such as temporal dynamics (e.g., periodicity and recency) and dynamic profit variations that have been playing an increasing role in many use cases nowadays.

There are some limitations which lead to redundant or less useful patterns in terms of making decisions in real time in dynamic environment scenarios. However, previous methods do not consider working altogether utility, time and profit factors and work quickly, which reduces their practicality.

IV. PROPOSED ALGORITHM

A. Objectives

Several important aims with the proposed DTU-Miner algorithm are as follows:

1. For effectively interweaving dynamic utility changes with temporal limitations (recency and periodicity).
2. Encouraging the use of state-of-the-art pruning methods to maximize computational effectiveness
3. To keep a lossless representation of high utility patterns for an intensive analysis.

B. CPR-Miner Algorithm [7]:

Algorithm 1 shows the steps of CPR-Miner that includes:

Step 1: It reads the dataset DB to calculate the Transaction Weighted Utility (TWU), Support (Sup), Maximum Period (MaxPer), and Recency (Re) for every item.

Step 2: Tt finds I*, which is the set of items that have their corresponding TWU, Sup, MaxPer, Re values above their specific thresholds μ, γ, $\lambda 2$, ρ.

Step 3: Items in I* are properly sorted by the ascending order of their TWU values.

Step 4: During the second scan of DB for each transaction, CPR-Miner filters out the items not belonging to I, then updates the remaining items in I of each transaction, further sorting the remaining items in each transaction in ascending order based on their updated TWU.

Step 5: In the third scan of DB, for every item i ∈ I, CPR-Miner builds the Utility-Period-Recency List (UPRL) for iii and the UPRS structure.

Step 6: The algorithm calls Search-CPRHUP (∅, ∅, I^, UPRLs, URSP) to find the entire collection of Closed Periodic Recent High Utility Patterns (CPRHUPs).
In this step, PPP and Pre-set(P) are first initialized to null while Post-set (PPP) is set to I.

Algorithm 1: CPR-Miner (CPRHUP)

Input: A quantitative database: DB; the levels: minimum

utility = μ, minimum recency = ρ, minimum period = minPer, max period = maxPer, minimum average

period = minAvg, max average period = maxAvg, and γ = |DB|/maxAvg − 1;
Output: All of the CPRHUPs.

1. Calculate the TWU, Sup, maxPer, Re of individual item as

scanning DB;
let $I* \subseteq I$ be such that their ∀twu, ∀sup, ∀maxPer, ∀Re ∈ J. μ, γ, $\lambda 2$, ρ, respectively;

2. Sort the elements in $I*$ with respect to ascending TWU values.
3. For each transaction, remove the items that are not found on the set $I*$ and increment the TWU of every item in $I*$ and order the leftovers starting from the lowest new TWU invented the number of items by each transaction;
4. Scan DB for constructing UPRLs of each item $i \in I*$ and the UPRS structure;
5. Call Search-CPRHUP (∅, ∅, $I*$, UPRLs, URSP);

C. The iEFIM-Closed algorithm

Algorithm 2 shows and describes the pseudo-code for iEFIM-Closed algorithm. It requires as input a transactional database and a user-defined threshold ξ. The final result which is returned are all the found Closed High Utility Itemsets (CHUIs), from the input database [8].
The calculation of P-set (X,i) and its application in function Search on lines 7 and 9 of Algorithm 2, respectively, are the main differences that are executed in iEFIM-Closed. The Search function (see Algorithm 2) uses P-set(X,i) in line 3, computes P-set(β,z) in line 6, and invokes Search recursively in line 12.

Since P-set(X,i) has involved all TIDs in projected database DX, the complexity of obtaining su(X,i)(support utility) in (7) of the main algorithm and (6) of Search function behaves the same as original EFIM-Closed algorithm.

The efficiency of P-set(X,i) can be seen in line 3 of the Search function, where it restricts its scanning to only those transactions that have respective TIDs in P-set(X,i), as opposed to scanning all transactions in the projected database DX. GZ-ET-EFIM-Closed employed this optimization to greatly reduce computational efficiency in comparison with pure EFIM-Closed.

Algorithm 2: The iEFIM-Closed algorithm (pseudo-code)

D − Transaction database D, ξ − The user-specific minimum utility threshold
Output: Collection of all closed high utility itemsets
1: X = ∅;
2: For all items i ∈ I scan D to compute lu(X, i);
3: Sec(X)={i|i∈I∧lu(X,i)≥ξ};
4: Sort Sec(X) from non-decreasing to decreasing according to lu(X, i);
5: Look for D to clean i∉ Sec(X) from transactions, discard empty transactions;
6: Read backwards and sort all transactions in D in non-decreasing order of lu(X, i);
7: Scan D to get su(X, i) and P-set(X, i), i ∈ Sec(X).
8: Pri(X) = {i|i ∈ Sec(X) ∧ su(X, i)≥ ξ};
9: RETURN Search: X, D, Pri(X), Sec(X), ξ, P-set(X, i);

D. Proposed Algorithm DTU-Miner

Therefore, combining the two, we developed an efficient mining framework, Algorithm 3 show the DTU-Miner, to mine Closed Periodic Recent High Utility Patterns (CPRHUPs) under both periodicity and recency temporal constraints with the dynamic profit adjustments. Here is the detailed description of the proposed algorithm:

All Utility, Time, and Dynamic Profit Factors are aggregated into the mining process. Leverage advanced pruning techniques to optimize computational efficiency. Extract patterns that are concise, lossless and actionable.

V. EXPERIMENTAL STUDIES

A. Dataset Description

To evaluate the performance and effectiveness of the proposed DTU-Miner Algorithm, several benchmark datasets from various fields were utilized. These datasets, summarized in Table III, vary in terms of size, density, and temporal characteristics, ensuring a comprehensive and balanced assessment of the algorithm's capability across different data types. Prior to analyzing the datasets in detail, it is useful to understand their general features. The selected benchmark datasets possess diverse density levels and dynamic profit values, making them suitable for testing under varying data conditions. Additionally, temporal attributes have been incorporated into these datasets, allowing the evaluation of how periodicity and recency constraints influence the algorithm's mining process. This setup ensures robust testing across multiple dimensions of data complexity.

Algorithm 3: DTU-Miner

Algorithm: DTU-Miner (Database D, MinUtil, MinPer, MaxPer, MinRec, MaxRec):
Input: D — Historical data transaction data base
MinUtil – The minimum utility threshold
Returns or Error meLess, meGreater - Returns or Error meLess and meGreater
MinRec, MaxRec - Minimum and Maximum recency thresholds
Output: R — List of high utility patterns
1. Set of result R ← ∅
2. Preprocess the Database D:
Calculate TWU (Transaction Weighted Utility) for every item.
3. Create initial utility-list structure (UPR-List):
For each item i in D:
Build UPR-List(i) as:
Actual Utility (AU)
Generalized Utility (GU)
Periodicity (Pe)
Recency (Re)
4. Input: a list UPR-List of patterns X
a. If TWU(X) < MinUtil, prune X.
b. Compute:
Utility (AU(X))
Periodicity metrics: MinPer(X), MaxPer(X)
Recency value: Rec(X)
c. If X meets the constraints:
AU(X) ≥ MinUtil
MinPer ≤ Pe(X) ≤ MaxPer
MinRec ≤ Re(X) ≤ MaxRec:
Then: R ← R U{X}
d. Generate candidate patterns by extending X.
5. Then apply pruning strategies to reduce candidates even further:
a. Dynamic Periodic Pruning (DPP)
b. PSP (Profit Stability Pruning)
c. Integrated Utility Pruning (IUP)
Remove the currently low utiltiy structures.
6. Return result set R.

TABLE III.
DATASET DESCRIPTION

Dataset Name	Type	T	Items	AIpT	Ch
Retail	Sparse	88,162	16,470	10	Real-world sales data
Mushroom	Dense	8,124	119	23	Dense transactional data
Chain Store	Sparse	1,112,949	46,086	7	Retail chain data
Foodmart	Dense	4,141	1,559	12	Grocery transactions

T: Transactions, AIpT: Avg. Items per Transaction,
Ch: Characteristics

B. Efficiency Evaluation

We evaluate efficiency of the proposed DTU-Miner algorithm through comparing with two state-of-the-art methods, namely CPR-Miner and iEFIM-Closed. The analysis is based on four key performance indicators:

- Runtime (in seconds)
- Memory Usage (in megabytes)
- Number of Extracted Patterns
- Pruning Rate (%) (as a percentage)

We evaluated our methods on four real datasets of different density, range and temporal patterns: Retail, Mushroom, Chain Store and Food Mart; referred to as Table III. The selected datasets in such an experimental setup are meant to real-world application, from sparse to very dense transactional structures to investigate the algorithmic robustness with dynamic profit-changing and temporal constraints. Compared with the traditional approaches, which lay stress on enormous calculation redundancy and frequent memory fragmentation, DTU-Miner utilizes the compact data structures (UPR-Lists and P-Sets) and efficient pruning methods. It significantly reduces run-time and memory usage, yet preserves high quality of patterns and its succinctness. Table IV. Shows the experimental results, and DTU-Miner performs the best consistently on all the datasets. Returning the lowest values for runtime and memory consumption, and the highest for pruning rate and level of relevance in the extracted patterns.

1) Runtime (s)

First, the study results illustrate that, based on Table IV, DTU-Miner attained superior runtime performance than CPR-Miner and iEFIM-Closed in terms of the datasets tested. In other words, the gap between the three algorithms' runtime performance progressively increased as the value of the tested datasets and database size used increased. Furthermore, the algorithm attained optimal runtime performance in the Chain Store and Mushroom datasets, recording a decline of over 20%. Hence, the study demonstrated that DTU-Miner optimized runtime performance through the developed pruning mechanisms, including the Dynamic Periodic Pruning, Profit Stability Pruning, and Integrated Utility Pruning. They are based on the concept of reducing the number of candidate patterns generated and minimizing unnecessary computations during the mining process.

2) Memory Usage (MB)

The DTU-Miner uses less memory to process all the datasets considered when compared with CPR-Miner and iEFIM-Closed. For example, on the Retail dataset, our DTU-Miner uses only 480 MB, and it is more efficient than both CPR-Miner and iEFIM-Closed which consume 560 MB and 530 MB respectively.
This efficiency is obtained by the use of small and well-structured data structures, like UPR-Lists and P-Sets that prevent the duplication of transactional knowledge and allow processing large populations without exceeding the memory capacity. DTU-Miner provides high accuracy with low memory overhead as it stores only the necessary pattern

statistics. This makes the algorithm especially favorable for usage in resource-limited systems or application in real-time.

3) *Patterns Extracted*

The DTU-Miner in general produces slightly fewer patterns than CPR-Miner and iEFIM-Closed. Although it may seem a limitation, this approach simply reflects the target of the algorithm to obtain the most representative patterns that indeed satisfy all the utility, periodicity and recency requirements. For instance, DTU-Miner mines 4,700 patterns in the Retail dataset while CPR-Miner and iEFIM-Closed obtain 4,512 and 4,200 patterns, respectively. However, the patterns generated by DTU-Miner are more concise, non-redundant, and interpretable, which is important for their application in real-world decision making problems.

4) *Pruning Rate (%)*

The DTU-Miner obtains the best pruning rate for all datasets, which can be up to 90% for Foodmart, and average pruning rate is kept much higher than CPR-Miner and iEFIM-Closed. The high pruning threshold is mainly achieved by combining a set of pruning techniques like Dynamic Periodic Pruning, Profit Stability Pruning and Integrated Utility Pruning, to kill non-promising candidates shortly at beginning of the mining process. To be able to drastically cut down the search space and still maintain high pattern quality, DTU-Miner is not only faster to mine, but it also better handles the overall accuracy and scalability.

C. *Observations*

According to the comparative results in Table V. We can make several observations on the performance of DTU-Miner in comparison with CPR-Miner and iEFIM-Closed as follows:

- Efficiency: DTU-Miner consistently outperforms in terms of runtime and memory efficiency in all the datasets. That is because of optimized data structures and pruning techniques which avoid unnecessary computations.
- Scalability: Our algorithm exhibits scalable performance and can handle both large-scale sparse datasets (e.g., Chain Store) and dense datasets (e.g., Mushroom) while steadily providing performance advantages.
- Relevance of Pattern: While at times DTU-Miner extracts fewer number of patterns but these patterns are less cumbersome and meaningful and are in accordance with the utility, periodicity and recency requirements. This choice of the active secondary extraction makes the results more comprehensible and applicable in the real case studies.
- Pruning: DTU-Miner has the highest pruning efficiencies compared to all compared algorithms, frequently above 85–90% and leading to a reduced computational cost and higher quality of the solutions.

The above observations support that DTU-Miner is not only more efficient but also more effective in extracting high-utility patterns for dynamic and temporally rich transactions.

TABLE IV.
EFFICIENCY EVALUATION

Dataset	Algorithm	Runtime (s)	Memory Usage (MB)	Patterns Extracted	Pruning Rate (%)
Retail	CPR-Miner	210	560	4,512	82
	iEFIM-Closed	230	530	4,200	80
	DTU-Miner	190	480	4,700	85
Mushroom	CPR-Miner	90	450	3,210	87
	iEFIM-Closed	105	420	3,050	85
	DTU-Miner	75	400	3,300	89
Chain Store	CPR-Miner	450	750	5,000	78
	iEFIM-Closed	480	700	4,850	76
	DTU-Miner	400	650	5,300	80

TABLE V.
PERFORMANCE COMPARISON OF ALGORITHMS

Metric	DTU-Miner	CPR-Miner	iEFIM-Closed
Efficiency	High efficiency due to advanced pruning and optimized structures.	Low efficiency; lacks robust pruning and compact structures.	Moderate efficiency; better than CPR-Miner but less optimized.
Scalability	Excellent scalability; performs well on large, dense, and sparse datasets.	Limited scalability; struggles with dense datasets.	Moderate scalability; better handling of dense datasets but limited in dynamic ones.
Accuracy and Pruning Rate (%)	High accuracy with the highest pruning rate (~85%). Filters non-promising patterns effectively.	Low pruning rate (~60%); produces redundant and less relevant patterns.	Moderate pruning rate (~75%); more effective than CPR-Miner but less precise than DTU-Miner.

VI. Conclusion

This paper introduces the DTU-Miner algorithm for mining closed high utility patterns which combines the dynamic adjustment of profit and temporal constraints, including periodicity and recency. DTU-Miner overcomes some limitations of the state of the art algorithms such as CPR-Miner and iEFIM-Closed, by using some advanced data structures called UPR-Lists and P-Sets and some new pruning techniques.

Empirical evaluations show that DTU-Miner excels in runtime, memory, and pruning power against state-of-the-art algorithms. This results in better scalability and accuracy on more general datasets, including generalized profit-based datasets with temporal constraints. Such an ability to discover short, lossless, and action invalid patterns shows the applicability of the usage of DTU-Miner in real-world databases of dynamic nature.

Future work may improve the adaptability of the algorithm even more to incremental datasets and may utilize the movement in domains such as e-commerce, healthcare, and supply chain analytics. Furthermore, exploring automation techniques for selecting threshold parameters could benefit the usability of the DTU-Miner algorithm.
Overall, DTU-Miner is a promising contribution to high utility pattern mining, offering a powerful and efficient method for discovering meaningful patterns in high utility data.

ACKNOWLEDGMENTS

We would like to thank the Southern Technical University, University of Basrah, Southern Technical University, and Independent Researcher, Shinjuku, incredibly if their assistance and cooperation. Their support and cooperation played a great role in the development and completion of this research.

Conflict of Interest

The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] J. M. Luna, P. Fournier‐Viger, and S. Ventura, "Frequent itemset mining: A 25 years' review ," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 9, no. 6, e1329, 2019, https://doi.org/10.1002/widm.1329

[2] B. Abdullah, and Vb. Narasimha, "High Utility Pattern Mining: A Survey on Current and Possible Areas of Applications," *Review of Information Engineering and Applications*, vol.9, no.1, pp. 38-49, 2022, https://doi.org/10.18488/79.v9i1.3236

[3] P. Fournier-Viger, C. W. Wu, S. Zida, and V. S. Tseng, "FHM: Faster high-utility itemset mining using estimated utility co-occurrence pruning," *in Foundations of Intelligent Systems: Proc. 21st Int. Symp*. ISMIS 2014, Roskilde, Denmark, vol. 8502, pp. 83–92, 2014, https://doi.org/10.1007/978-3-319-08326-1_9

[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: Proc.* 9th Pacific-Asia Conf. PAKDD 2005, Hanoi, Vietnam, vol. 3518, pp. 689–695, 2005, https://doi.org/10.1007/11430919_79

[5] A. A. Al-Hamodi and S. F. Lu, "MapReduce Frequent Itemsets for Mining Association Rules," *in Proc. Int. Conf. Information System and Artificial Intelligence (ISAI)*, pp. 281–284, IEEE, 2016, https://doi.org/10.1109/ISAI.2016.0066

[6] A. A. Al-Hamodi and S. Lu, "MRFP: discovery frequent patterns using MapReduce frequent pattern growth," *in Proc. Int. Conf. Network and Information Systems for Computers (ICNISC)*, pp. 298–301, IEEE, 2016, https://doi.org/10.1109/ICNISC.2016.071

[7] Y. Qi, X. Zhang, G. Chen, and W. Gan, "Mining periodic trends via closed high utility patterns," *Expert Systems with Applications*, vol. 228, 2023, https://doi.org/10.1016/j.eswa.2023.120356

[8] T. D. Nguyen et al., "Efficient algorithms for mining closed high utility itemsets in dynamic profit databases," *Expert Systems with Applications*, vol. 186, 2021, https://doi.org/10.1016/j.eswa.2021.115741

[9] M. Liu and J. Qu, "Mining high utility itemsets without candidate generation," *in Proc. 21st ACM Int. Conf. Information and Knowledge Management*, pp. 55–64, 2012, https://doi.org/10.1145/2396761.2396773

[10] Y. Liu, W.-K. Liao, and A. Choudhary, "A two-phase algorithm for fast discovery of high utility itemsets," *in Proc. Pacific-Asia Conf. Knowledge Discovery and Data Mining,* pp. 689–695, 2005, https://doi.org/10.1007/11430919_79

[11] P. Fournier-Viger, S. Zida, J. C.-W. Lin, C.-W. Wu, and V. S. Tseng, "EFIM-closed: Fast and memory efficient discovery of closed high-utility itemsets," *in Proc. Int. Conf. Machine Learning and Data Mining in Pattern Recognition*, vol. 9729, pp. 199–213, 2016, https://doi.org/10.1007/978-3-319-41920-6_15

[12] V. S. Tseng, C.-W. Wu, B.-E. Shie, and P. S. Yu, "UP-Growth: An efficient algorithm for high utility itemset mining," *in Proc. 16th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, pp. 253–262, 2010, https://doi.org/10.1145/1835804.1835839

[13] S. K. Tanbeer, C. F. Ahmed, B. S. Jeong, and Y. H. Lee, "Discovering periodic frequent patterns in transactional databases," *in Proc. 13th Pacific-Asia Conf. Knowledge Discovery and Data Mining, LNCS,* vol. 5476, pp. 242–253, 2009, https://doi.org/10.1007/978-3-642-01307-2_24

[14] K. Amphawan, P. Lenca, and A. Surarerks, "Mining Top-K periodic-frequent pattern from transactional databases without support threshold," *in Proc. 3rd Int. Conf. Advanced Information Technology, CCIS,* vol. 55, pp. 18–29, 2009, https://doi.org/10.1007/978-3-642-10392-6_3

[15] D. T. Dinh, B. Le, P. Fournier-Viger, and V. N. Huynh, "An efficient algorithm for mining periodic high-utility

sequential patterns," *Applied Intelligence*, vol. 48, no. 12, pp. 4694–4714, 2018, https://doi.org/10.1007/s10489-018-1227-x

[16] V. S. Tseng, C.-W. Wu, P. Fournier-Viger, and P. S. Yu, "Efficient algorithms for mining the concise and lossless representation of high utility itemsets," *IEEE Trans. on Knowledge and Data Engineering,* vol. 27, no. 3, pp. 726–739, 2014, https://doi.org/10.1109/TKDE.2014.2345377

[17] T.-L. Dam, K. Li, P. Fournier-Viger, and Q.-H. Duong, "CLS-Miner: Efficient and effective closed high-utility itemset mining," *Frontiers of Computer Science*, vol. 13, no. 2, pp. 357–381, 2019, https://doi.org/10.1007/s11704-016-6245-4

[18] A. A. Ghaib, Y. E. A. Alsalhi, I. M. Hayder, H. A. Younis, and A. A. Nahi, "Improving the Efficiency of Distributed Utility Item Sets Mining in Relation to Big Data," *Journal of Computer Science and Technology Studie*s, vol. 5, no. 4, pp. 122–131, 2023, https://doi.org/10.47760/ijcsmc.2024.v13i01.003

[19] A. A. Ghaib and A. A. Nahi, "Enhancing N-List Structure and Performance for Efficient Large Dataset Analysis," *Int. J. of Computer Science and Mobile Computing*, vol. 13, no. 1, pp. 49–58, 2024, https://doi.org/10.47760/ijcsmc.2024.v13i01.003

[20] B. A. Shtayt, N. H. Zakaria, and N. H. Harun, "A comprehensive review on medical image steganography based on LSB technique and potential challenges," *Baghdad Science Journal*, vol. 18, no. 2 Suppl., pp. 0957–0957, 2021, https://doi.org/10.21123/bsj.2021.18.2(Suppl.).0957

[21] A. A. Ghaib, "DU-HUIM: A novel dynamic utility high-utility itemset miner algorithm for dynamic utility variations in big data," *IRJET*, vol. 12, no. 01, pp. 405–410, 2025.

[22] D. Gurakuq. "Analytical analysis of electromagnetic torque and magnet utilization factor for two different PM machines with SPM and HUPM rotor topologies," *IEEE Transactions on Magnetics* vol. 57, no. 6 pp. 1-9, 2021, https://doi.org/10.1109/TMAG.2021.3069082