Weighted frequent itemset mining over uncertain databases

Article *in* Applied Intelligence · August 2015

DOI: 10.1007/s10489-015-0703-9

CITATIONS
60

READS
321

5 authors, including:

Wensheng Gan
Jinan University (Guangzhou, China)
220 PUBLICATIONS **3,400** CITATIONS

SEE PROFILE

Philippe Fournier Viger
Shenzhen University
428 PUBLICATIONS **11,991** CITATIONS

SEE PROFILE

Tzung-Pei Hong
National University of Kaohsiung
794 PUBLICATIONS **12,612** CITATIONS

SEE PROFILE

Appl Intell (2016) 44:232–250
DOI 10.1007/s10489-015-0703-9

Weighted frequent itemset mining over uncertain databases

Jerry Chun-Wei Lin[1] · Wensheng Gan[1] · Philippe Fournier-Viger[2] · Tzung-Pei Hong[3,4] · Vincent S. Tseng[5]

Published online: 8 August 2015

Abstract Frequent itemset mining (FIM) is a fundamental research topic, which consists of discovering useful and meaningful relationships between items in transaction databases. However, FIM suffers from two important limitations. First, it assumes that all items have the same importance. Second, it ignores the fact that data collected in a real-life environment is often inaccurate, imprecise, or incomplete. To address these issues and mine more useful and meaningful knowledge, the problems of weighted and uncertain itemset mining have been respectively proposed, where a user may respectively assign weights to items to specify their relative importance, and specify existential probabilities to represent uncertainty in transactions. However, no work has addressed both of these issues at the same time. In this paper, we address this important research problem by designing a new type of patterns named high expected weighted itemset (HEWI) and the HEWI-Uapriori algorithm to efficiently discover HEWIs. The HEWI-Uapriori finds HEWIs using an Apriori-like two-phase approach. The algorithm introduces a property named high upper-bound expected weighted downward closure (HUBEWDC) to early prune the search space and unpromising itemsets. Substantial experiments on real-life and synthetic datasets are conducted to evaluate the performance of the proposed algorithm in terms of runtime, memory consumption, and number of patterns found. Results show that the proposed algorithm has excellent performance and scalability compared with traditional methods for weighted-itemset mining and uncertain itemset mining.

Keywords Data mining · Uncertain databases · Weighted frequent itemsets · Two-phase · Upper-bound

✉ Jerry Chun-Wei Lin
jerrylin@ieee.org

Wensheng Gan
wsgan001@gmail.com

Philippe Fournier-Viger
philippe.fournier-viger@umoncton.ca

Tzung-Pei Hong
tphong@nuk.edu.tw

Vincent S. Tseng
vtseng@cs.nctu.edu.tw

[1] School of Computer Science and Technology, Harbin Institute of Technology Shenzhen Graduate School, Shenzhen 518055, People's Republic of China

[2] Department of Computer Science, University of Moncton, Moncton, Canada

[3] Department of Computer Science and Information Engineering National University of Kaohsiung, Kaohsiung, ROC, Taiwan

[4] Department of Computer Science and Engineering, National Sun Yat-sen University, Kaohsiung, ROC, Taiwan

[5] Department of Computer Science, National Chiao Tung University, Hsinchu, ROC, Taiwan

1 Introduction

In the field of Knowledge Discovery in Databases (KDD), frequent itemset mining (FIM) [9, 12] is a fundamental task, which consists of discovering groups of items (itemsets) frequently appearing together in transaction databases. FIM has numerous real-life applications and is crucial to

or has inspired several data mining tasks such as association rule mining [5, 12], sequential pattern mining [20, 21], pattern mining with interesting measures [11, 18], and constraint-based frequent pattern mining [17, 31]. FIM has been extensively studied. However, it can be observed that all studies on FIM suffer from at least one of two limitations.

First, most studies ignore the fact that data collected in real-life applications such as wireless sensor network or location based-service may be inaccurate, imprecise or incomplete. To address this issue, it is necessary to study data uncertainty [3, 4]. Thus, developing efficient algorithms to mine patterns in uncertain databases has become a major research topic in recent years. Many algorithms have been developed to mine frequent itemsets in uncertain databases [2, 4, 7, 10, 15, 19, 23, 25]. These algorithms may be generally classified into two categories: those based on the expected support model [2, 10, 15, 19] and those based on the probabilistic frequent itemset mining model [7, 22]. To mine frequent itemsets in an uncertain database, a minimum expected support threshold has to be specified by the user. This threshold is used as a constraint to determine which itemsets shall be found.

Second, most studies assume binary occurrences of items in all transactions, i.e. that all items are either present or absent in each transaction, and they all have the same importance. But in real-world applications, not all items are equally important for the user. To address this issue, a prominent solution is to let the user assign weights to items to indicate their relative importance (e.g. interestingness, importance, profit or risk). In traditional FIM, only the absolute or relative frequency of itemsets (aka support count) is considered to identify patterns. However, the support is an insufficient measure to identify useful and meaningful patterns because it does not take the importance or interestingness of items into account. Thus, itemsets that are not frequent but have a high importance may be discarded during the mining process. Facing this issue, the problem of mining weighted frequent itemsets has been proposed and extensively studied [8, 14, 24, 26–30], where both the importance and occurrence frequencies of itemsets are considered. In this context, the weight of a pattern represents its importance, interest, or risk which can be defined based on a user's preferences or domain knowledge. For example, let be two frequent itemsets (AC) and (CE) found in a database. If their weights are respectively 0.9 and 0.1, itemset (AC) is considered more important and useful than itemset (CE). Using weights allows pruning the search space very efficiently. Furthermore, an advantage of weighted itemset mining over traditional FIM is that it discovers fewer itemsets but itemsets having a high importance. However, a crucial challenge to design efficient weighted pattern mining algorithms is that the powerful anti-monotone property used in FIM to prune the search space generally does not hold when different weights are assigned to items. In other words, an infrequent itemset may have supersets that are either frequent or infrequent when different weights are considered. To restore the anti-monotone property for mining weighted frequent patterns, the maximal weight of all items in the transactional database is generally used [28].

Given that a database may contain inaccurate or uncertain information (for example, if the data has been collected from a sensor network), it is a critical issue to develop an efficient approach for mining the weighted frequent itemsets in uncertain databases. However, a major challenge is that if we consider the weights of items, the downward closure property used for mining frequent itemsets in uncertain databases may not hold anymore. As a result, itemsets having low weights may be output or their weights may be incorrectly calculated. Moreover, previous uncertain itemset mining algorithms cannot be applied to mine the weighted frequent itemsets in uncertain databases. The main challenge of mining weighted frequent itemsets in uncertain databases is to utilize the downward closure property, to be able to prune the search space. Besides, a benefit of mining weighted frequent itemsets in uncertain databases is to discover more useful and meaningful knowledge for decision making.

In this paper, we address the above challenges by proposing the problem of high expected weighted itemset (HEWI) mining, to discover weighted frequent itemsets in uncertain databases. Each item in an uncertain database has a weight representing its importance, and each item appears in each transaction with an existential probability. Based on the designed representation of HEWIs, more meaningful and useful information can be discovered because both the weight and uncertainty constraints are considered. The major contributions of this paper are summarized as follows:

1. A new type of patterns called high expected weighted itemset (HEWI) is developed to consider both the weights and existential probabilities of items, and thus to discover more meaningful and useful patterns in uncertain databases.
2. Based on the concept of HEWI, a weight upper-bound and a downward closure property namely high upper-bound expected weighted downward closure (HUBEWDC) are developed to early prune the search space and unpromising itemsets.
3. The HEWI-Uapriori algorithm is proposed to discover HEWIs using the HUBEWDC in two phases. In the first phase, an Apriori-like approach is performed to discover the designed high transaction expected weighted itemsets (HTEWIs) in a level-wise manner. In the second phase, an additional database scan is performed to identify the actual HEWIs from the set of HUBEWIs.

4. Substantial experiments indicate that the proposed HEWI-Uapriori algorithm can be used to efficiently mine more useful and meaningful HEWIs compared to the traditional algorithms for weighted frequent itemset mining in a binary database or for expected support frequent itemset mining in an uncertain database.

The rest of this paper is organized as follows. Related work is reviewed in Section 2. The preliminaries and problem statement associated with the designed HEWIs are given in Section 3. The proposed HEWI-Uapriori algorithm is described in Section 4. Section 5 is an illustrated example of the HEWI-Uapriori algorithm. Results of an experimental evaluation comparing the performance of the proposed algorithm with traditional algorithms for mining the weighted frequent itemsets in a binary database and the expected support frequent itemsets in an uncertain database, is provided in Section 6. Finally, a conclusion is drawn and future work are discussed in Section 7.

2 Related work

In this section, related work about frequent itemset mining in uncertain databases and weighted frequent pattern mining in binary databases are briefly reviewed.

2.1 Frequent itemset mining in uncertain databases

Mining frequent itemsets is a fundamental task in data mining to discover interesting relationships between items in a database. Frequent itemset mining (FIM) only considers whether an item or itemset is present or absent in a transactional database. But in real-life applications, the huge amount of data stored in a database may be inaccurate, imprecise, or incomplete (e.g. if the data has been obtained from a wireless sensor network or location-based services [3, 4]). To address this issue, it is necessary to study data uncertainty. Thus, developing efficient algorithms to mine patterns in uncertain databases has become a major research topic in recent years, which has many applications [2, 4, 7, 10, 15, 19, 23, 25]. Algorithms for mining uncertain frequent itemsets in uncertain databases can be generally classified into two categories: those based on the expected support model [2, 10, 15, 19] and those based on the probabilistic frequent itemset mining model [7, 22]. The expected support model estimates the support of itemsets based on existential probabilities, while the probabilistic frequent model uses the frequentness probability as a measure to mine the probabilistic frequent itemsets. According to the expected support model, an itemset is considered frequent in an uncertain database if its expected support ratio is no less than a minimum expected support threshold. In the probabilistic frequent model, an itemset is considered frequent if its frequentness probability is no less than a specified minimum probability.

Chui et al. first introduced the problem of frequent itemset mining in uncertain databases. They proposed the Uapriori algorithm, which mines frequent itemsets in uncertain databases using a level-wise approach inspired by the well-known Apriori algorithm [10]. The expected support threshold measure was defined to mine frequent itemsets in uncertain databases. Leung et al. designed the UFP-growth algorithm [15] to mine uncertain frequent itemsets using an extended FP-tree structure and a divide-and-conquer and depth-first search approach. The UFP-growth algorithm was shown to be much faster for discovering uncertain frequent itemsets than the Uapriori algorithm. Aggarwal et al. then presented the UH-mine algorithm [2] to efficiently mine frequent itemsets based on an extended H-Mine approach, using a structure called UH-Struct. UH-mine also adopts a divide-and-conquer and depth-first search approach to recursively discovers frequent itemsets. Experiments have shown that UH-mine considerably outperforms previous approaches for mining uncertain frequent itemsets. Lin et al. also proposed a tree-based CUFP-growth algorithm to mine frequent itemsets from a compressed uncertain frequent pattern (CUFP)-tree structure [19]. All the aforementioned algorithms such as Uapriori [10], UFP-growth [2], UH-mine [2] and CUFP-growth [19] belong to the expected support model.

In addition to the expected support model, Bernecker et al. proposed a novel probabilistic formulation for mining frequent itemsets in uncertain database based on possible world semantics [7]. Sun et al. also developed the p-FP structure and proposed the bottom-up (p-Apriori) and top-down (TODIS) algorithms to discover frequent patterns in an uncertain database [22]. The p-Apriori and TODIS algorithms adopt a dynamic programing and divide-and-conquer approach for mining probabilistic frequent itemsets based on the probabilistic frequent model.

Besides algorithms for mining frequent itemsets in uncertain databases, some algorithms have been proposed for mining frequent itemsets in uncertain data streams such as UF-streaming [16] and SUF-growth [16]. Mining weighted frequent itemsets in uncertain databases by concerning both the weight and uncertain factors has however, not been proposed yet.

2.2 Weighted frequent pattern mining in binary databases

Weighted frequent pattern mining in binary databases is a generalization of the problem of frequent pattern mining,

where weights are assigned to each item by the user to indicate their relative importance or interestingness. Because weights are considered during the mining process, more useful and interesting patterns can be discovered according to users' preferences. In recent years, numerous algorithms have been developed for weighted frequent pattern mining [8, 14, 24, 26–30], for example, to discover weighted frequent itemsets [28], weighted association rules [8, 24, 27] and weighted sequential patterns [14, 29, 30].

Cai et al. proposed the weighted-support measure to keep the anti-monotone property in association rule mining when weighted items are considered. This measure consists of multiplying an itemset's support by the average weight of the items in the itemset [8]. Wang et al. proposed to mine Weighted Association Rules (WAR) [27] by first generating frequent itemsets without considering item weights and then deriving weighted association rules from frequent itemsets using the designed ordered shrinkage approach. Tao et al. then proposed the Weighted Association Rule Mining (WARM) algorithm to mine WAR by considering weights during the iterative process of frequent itemset generation [24]. The problem of invalidation the downward closure property of the support when considering weights was solved by the "weighted downward closure" property. The Weighted Frequent Itemset Mining (WFIM) algorithm [28] was proposed to consider the weight constraint when mining itemsets using a pattern-growth approach. WFIM used the weight range and the minimum weight strategies to maintain the downward closure property. Conducted experiments have shown that WFIM can generate more concise and important weighted frequent itemsets. Yun et al extended the idea of using weights in pattern mining to find weighted sequential patterns. They proposed the efficient WSpan algorithm to mine weighted sequential patterns [29]. WSpan applied the maximal weight among all items as the maximum weight of each sequence to discover weighted sequential patterns.

Using the weight constraint in pattern mining was shown to provide more interesting patterns to users because weights assigned according to users' preferences are considered during the mining process. However, developing algorithms for weighted pattern mining is still a very active research topic. To our knowledge, mining weighted frequent itemsets in uncertain databases has never been explored.

3 Preliminaries and problem statement

This section first introduces preliminaries and then defines the proposed problem of mining high expected weighted itemsets (HEWIs) in uncertain databases.

3.1 Preliminaries

Let $I = \{i_1, i_2, \ldots, i_m\}$ be a finite set of m distinct items. An uncertain database is a set of transactions $D = \{T_1, T_2, \ldots, T_n\}$, where each transaction $T_q \in D$ is a subset of I, and has an unique identifier, called its *TID*. Based on the attribute uncertainty database model [10], an unique existence probability $p(i_j, T_q)$ is assigned to each item i_j in each transaction T_q. It indicates that an item i_j exists in T_q with a probability $p(i_j, T_q)$. Moreover, a weight table is defined as $wtable = \{w(i_1), w(i_2), \ldots, w(i_m)\}$, where $w(i_j)$ is the weight of an item i_j. An itemset X is a set of k distinct items $\{i_1, i_2, \ldots, i_k\}$, such that X is said to be a $k-$itemset, and k is the length of the itemset. An itemset X is contained in a transaction T_q if $X \subseteq T_q$. A predefined minimum expected weighted-support threshold is defined as $\varepsilon \in (0, 100\ \%]$. An example database defined according to the attribute uncertainty model is shown in Table 1 and will be used as the running example. It consists of 10 transactions and 6 items denoted from (A) to (F).

In this example, we will assume that the minimum expected weighted-support threshold ε is set to 10% and that item weights are defined according to the weight (w)-table shown in Table 2.

Definition 1 (Item weight) The weight of an item i_j in a database D indicates the item importance and is denoted as $w(i_j)$, where $w(i_j) \in (0, 1]$.

In the given example of Table 2, the weight of (A) and (E) are respectively $w(A)$ $(= 0.2)$ and $w(E)$ $(= 0.55)$

Definition 2 (Itemset weight in a transaction) The weight of an itemset X in a transaction T_q is denoted as $w(X, T_q)$, where $w(X, T_q)$ is the sum of the weights of all items in

Table 1 An uncertain database

TID	Transaction (item, probability)
1	$(A, 0.25)$, $(C, 0.4)$ $(E, 1.0)$
2	$(D, 0.35)$ $(F, 0.7)$
3	$(A, 0.7)$ $(B, 0.82)$ $(C, 0.9)$ $(E, 1.0)$ $(F, 0.7)$
4	$(D, 1.0)$ $(F\ 0.5)$
5	$(B, 0.4)$ $(C, 0.4)$, $(D, 1.0)$
6	$(A, 0.8)$ $(B, 0.8)$ $(C, 1.0)$ $(F\ 0.3)$
7	$(B, 0.8)$ $(C, 0.9)$ $(D, 0.5)$ $(E, 1.0)$
8	$(B, 0.65)$ $(E\ 0.4)$
9	$(B, 0.5)$ $(D, 0.8)$ $(F, 1.0)$
10	$(A, 0.4)$ $(B, 1.0)$ $(C, 0.9)$ $(E\ 0.85)$

Table 2 Weight (w)-table

Item	*A*	*B*	*C*	*D*	*E*	*F*
Weight	0.2	0.75	0.9	1.0	0.55	0.3

X divided by the number of items in X, which is formally defined as:

$$w(X, T_q) = \frac{\sum_{i_j \in X} w(i_j, T_q)}{|k|}$$

where $|k|$ is the number of items in X, and $w(i_j, T_q)$ is equal to $w(i_j)$.

In the example of Tables 1 and 2, the weights of (C) and (ACE) in transaction T_1 are respectively calculated as $w(C, T_1) = w(C, T_1)/1$ $(= 0.9/1)$ $(= 0.9)$, and $w(ACE, T_1) = (w(A, T_1) + w(C, T_1) + w(E, T_1))/3$ $(= (0.2 + 0.9 + 0.55)/3)$ $(= 0.55)$.

Definition 3 (Itemset weight in D) Given that the weight of an itemset X is the same for all transactions of a database D, the weight of an itemset X in a database D is denoted as $w(X)$, and is defined equal to $w(X, T_q)$ for any transaction T_q containing X, i.e. $w(X) = w(X, T_q)$.

In the example of Tables 1 and 2, the weights of (C) and (ACE) in the database are respectively calculated as $w(C) = w(C, T_1) = w(C, T_3) = w(C, T_5) = w(C, T_6) = w(C, T_7) = w(C, T_{10})$ $(= 0.9/1)$ $(= 0.9)$, and $w(ACE) = w(ACE, T_1) = w(ACE, T_3) = w(ACE, T_{10}) = (0.2 + 0.9 + 0.55)/3$ $(= 0.55)$.

Definition 4 (Itemset probability in a transaction) The probability of an itemset X in a transaction T_q is denoted as $p(X, T_q)$, and is defined as:

$$p(X, T_q) = \prod_{i_j \in X} p(i_j, T_q),$$

where j is the j-th item in X.

Consider the example of Table 1. The probabilities of item (C) and itemset (ACE) in transaction T_1 are respectively calculated as: $p(C, T_1)$ $(= 0.4)$ and $p(ACE, T_1) = p(A, T_1) \times p(C, T_1) \times p(E, T_1) = 0.25 \times 0.4 \times 1.0$ $(= 0.1)$.

Based on the expected support model used in uncertain data mining, the expected support of an itemset X in a database D is computed by summing the support of X in all possible worlds.

Definition 5 (Expected support of an itemset in ***D***) The expected support of an itemset X in an uncertain database D is denoted as expSup(X), and is defined as the sum of the expected probabilities of X in the transactions where X appears, thus:

$$expSup(X) = \sum_{X \subseteq T_q \wedge T_q \in D} p(X, T_q) = \sum_{X \subseteq T_q \wedge T_q \in D} \left(\prod_{i_j \in X} p(i_j, T_q) \right).$$

For example, in Table 1, $expSup(A) = p(A, T_1) + p(A, T_3) + p(A, T_6) + p(A, T_{10}) = 0.25 + 0.7 + 0.8 + 0.4$ $(= 2.15)$, and $expSup(ACE) = p(ACE, T_1) + p(ACE, T_3) + p(ACE, T_{10}) = 0.1 + 0.63 + 0.306$ $(= 1.036)$.

Definition 6 (Expected weighted support of an itemset in D) The expected weighted support of an itemset X in an uncertain database D is denoted as $expWSup(X)$, and is defined as the weight of X multiplied by the expected support of X, that is:

$$expWSup(X) = w(X) \times expSup(X) = w(X) \times \sum_{X \subseteq T_q \wedge T_q \in D} p(X, T_q).$$

In the example of Tables 1 and 2, the expected weighted supports of itemset (A) and (ACE) are respectively calculated as $expWSup(A) = w(A) \times expSup(A) = 0.2 \times 2.15$ $(= 0.43)$ and $expWSup(ACE) = w(ACE) \times expSup(ACE) = 0.55 \times 1.036$ $(= 0.5698)$.

Definition 7 (High expected weighted itemset, HEWI) An itemset X is said to be a high expected weighted support itemset (HEWI) in an uncertain database D if the expected weighted support of X in D is no less than the minimum expected weighted support (the minimum expected weighted support threshold ε multiplied by the number of transactions in D), that is if $expWSup(X) \geq \varepsilon \times |D|$.

Consider the example of Table 1 and that the minimum expected support threshold ε is set to 10%. Table 3 shows the expected frequent itemsets found in the uncertain database when weights are not considered (i.e. when all item weights are set to 1). The expected weighted supports (*expSup*) of itemsets are shown in the third column of Table 3. The fourth column shows the expected weighted support when weights of Table 2 are considered.

From Table 3, it can be seen that only considering the probability measure is insufficient for revealing interesting and useful patterns. For example, if the weight constraint is not considered, $expSup(A)$ $(= 2.15)$. However, if the weight constraint is considered, $expWSup(A)$ $(= 0.43)$. Therefore, when considering both the weight and existential probabilities, fewer and more useful patterns are found. For example in Table 3, when the minimum expected weighted-support

Table 3 Discovered itemsets when ignoring the weight constraint

Itemset	Weight	Frequency	*expSup*	*expWSup*
(*A*)	0.200	4	2.15	0.43
(*B*)	0.75	7	4.97	3.727
(*C*)	0.900	6	4.500	4.05
(*D*)	1.000	5	3.65	3.65
(*E*)	0.55	5	4.25	2.337
(*F*)	0.300	5	3.20	0.96
(*AB*)	0.475	3	1.614	0.766
(*AC*)	0.55	4	1.89	1.039
(*AE*)	0.375	3	1.29	0.484
(*BC*)	0.825	5	3.318	2.737
(*BD*)	0.875	3	1.200	1.05
(*BE*)	0.65	4	2.73	1.774
(*BF*)	0.525	3	1.314	0.690
(*CE*)	0.725	4	2.965	2.150
(*DF*)	0.65	3	1.545	1.000
(*ABC*)	0.617	3	1.517	0.935
(*ACE*)	0.55	3	1.036	0.570
(*BCE*)	0.733	3	2.223	1.63

threshold is set to 10%, the itemset (A) is not a high expected weighted itemset (HEWI) since $expWSup(A) = w(A) \times expSup(A)$ $(= 0.2 \times 2.15)$ $(= 0.43)$, which is less than the minimum expected weighted support $\varepsilon \times |D| =$ $(10\% \times 10)$ $(= 1.0)$. But the itemset (BCE) is a HEWI since $expWSup(BCE)$ $(= 1.63 > 1.0)$. Table 4 presents the set of discovered HEWIs when weights are considered.

3.2 Problem statement

Based on the above definitions, the problem of discovering the designed high expected weighted itemsets (HEWIs) in uncertain databases is formulated as follows:

Problem Statement Let be an uncertain database D, a user-specified weight table *wtable* and a user-specified minimum expected weighted-support threshold ε. The problem of mining high expected weighted itemsets (HEWIs) in an uncertain database D is to discover the set of expected weighted frequent k-itemsets while considering both the weight and the existential probability constraints. An itemset X is a HEWI if its expected weighted support $expWSup(X)$ is no less than the minimum expected weighted support, that is $expWSup(X) \geq \varepsilon \times |D|$

4 Proposed algorithm to mine the designed high expected weighted itemsets

In this section, an upper-bound related to the designed concept of high expected weighted itemsets (HEWIs) is first proposed, the downward closure property is than developed. Finally, a two-phase Apriori-like approach named HEWI-Uapriori is designed to mine HEWIs in an uncertain database.

4.1 Proposed upper-bound and downward closure property

Numerous algorithms have been proposed to mine the weighted frequent itemsets or the expected support frequent itemsets. In previous studies [2, 10, 15, 19] on expected support frequent itemset mining, the expected support downward closure property was proposed to reduce the search space, and thus speed up the process of mining the expected support frequent itemsets.

Theorem 1 (Expected support downward closure property) *If an itemset X is an expected support frequent itemset, any subset of X is also an expected support frequent itemset.*

Proof Let X^k denotes an itemset of length k, X^{k-1} denotes a subset of X^k of length $k-1$, and T_q be a transaction. Since $p(X, T_q) = \prod_{i_j \in X} p(i_j, T_q)$, it follows that:

$$\frac{p(X^k, T_q)}{p(X^{k-1}, T_q)} = \frac{\prod_{i_j \in X^k} p(i_j, T_q)}{\prod_{i_j \in X^{k-1}} p(i_j, T_q)} = \prod_{i_j \in (X^k - X^{k-1})} p(i_j, T_q) \leq 1$$

The above result indicates that X^{k-1} has a higher probability than X^k in the same transaction that is

Table 4 Discovered itemsets (HEWIs) when weights are considered

Itemset	Weight	Frequency	*expSup*	*expWSup*
(*B*)	0.75	7	4.97	3.727
(*C*)	0.900	6	4.500	4.05
(*D*)	1.000	5	3.65	3.65
(*E*)	0.55	5	4.25	2.337
(*AC*)	0.55	4	1.89	1.039
(*BC*)	0.825	5	3.318	2.737
(*BD*)	0.875	3	1.200	1.05
(*BE*)	0.65	4	2.73	1.774
(*CE*)	0.725	4	2.965	2.150
(*DF*)	0.65	3	1.545	1.000
(*BCE*)	0.733	3	2.223	1.63

$p(X^{k-1}, T_q) \geq p(X^k, T_q)$. Let $TID(X^k)$ and $TID(X^{k-1})$ respectively denote the list of TIDs of transactions containing X^k and X^{k-1}. Because $TID(X^{k-1}) \geq TID(X^k)$, it follows that:

$\sum_{T_q \in TID(X^{k-1})} p(X^{k-1}, T_q) \geq \sum_{T_q \in TID(X^k)} p(X^k, T_q)$
$\Rightarrow expSup(X^{k-1}) \geq expSup(X^k)$.

Then, it follows that if X^k is an expected support frequent itemset ($expSup(X^k) \geq$ minimum expected support), any subset X^{k-1} is also an expected support frequent itemset ($expSup(X^{k-1}) \geq$ minimum expected support). □

Several studies have addressed the problem of designing a downward closure property for weighted frequent itemset mining. However, from the results of Table 4, it can be observed that the well-known downward closure property used in association-rule mining [5] does not hold for mining high expected weighted itemsets (HEWIs) in an uncertain database. For example, the itemsets (*A*) and (*F*) are not HEWIs but their supersets (*AC*) and (*DF*) are HEWIs. If no downward closure property is used, a level-wise algorithm may generate a huge number of candidates for mining HEWIs. It is thus crucial to define a downward closure property for mining HEWIs that considers both the existential probabilities and weights of itemsets. In the proposed HEWI-Uapriori algorithm, a novel transaction upper-bound weighted probability (*tubwp*) is designed to maintain the downward closure property of HEWIs. This allows to greatly reduce the number of itemsets that need to be considered in the search space to mine the HEWIs.

Definition 8 (Transaction upper-bound weight, ***tubw***). The transaction upper-bound weight of a transaction T_q is denoted as $tubw(T_q)$, and is defined as:

$tubw(T_q) = max\{w(i_1 T_q), w(i_2, T_q), \ldots, w(i_j, T_q)\}$,

where $w(i_j, T_q)$ is equal to $w(i_j)$, and j is the number of items in T_q.

In the example of Table 1, $tubw(T_1) = max\{w(A, T_1), w(C, T_1), w(E, T_1)\} = max\{0.2, 0.9, 0.55\}$ (= 0.9). Moreover, the transaction upper-bound weight of each transaction is shown in Table 5.

Theorem 2 (***w(X, T_q) ≤ tubw(T_q)***) *The weight of any itemset X in a transaction T_q is always less than or equal to the transaction upper-bound weight of T_q, that is $w(X, T_q) \leq tubw(T_q)$.*

Proof Since $tubw(T_q) = max\{w(i_1, T_q), w(i_2, T_q), \ldots, w(i_j, T_q)\}$

$$w(X, T_q) = \frac{\sum_{i_j \in X \wedge X \subseteq T_q} w(i_j, T_q)}{|k|} \leq \frac{\max\{w(i_j, T_q)\} \times |k|}{|k|} = tubw(T_q)$$

Therefore, it follows that $w(X, T_q) \leq tubw(T_q)$. □

Definition 9 (Transaction upper-bound probability, ***tubp***) The transaction upper-bound probability of a transaction T_q is denoted as $tubp(T_q)$, and defined as:

$tubp(T_q) = \max\{p(i_1, T_q), p(i_2, T_q), \ldots, p(i_j, T_q)\}$

Table 5 Transaction upper-bound weight of each transaction

TID	1	2	3	4	5	6	7	8	9	10
tubw	0.9	1.0	0.9	1.0	1.0	0.9	1.0	0.75	1.0	0.9

Table 6 Transaction upper-bound probability of each transaction

TID	1	2	3	4	5	6	7	8	9	10
tubp	1.0	0.7	1.0	1.0	1.0	1.0	1.0	0.65	1.0	1.0

where j is the number of items in T_q.

For example, $tubp(T_1) = max\{p(A, T_1), p(C, T_1), p(E, T_1)\} = max\{0.25, 0.4, 1.0\} (= 1.0)$. Moreover, the transaction upper-bound probability of each transaction is shown in Table 6.

Theorem 3 ***(p(X,T$_q$) ≤ tubp(T$_q$))*** *The probability of any itemset X in a transaction T_q is always less than or equal to the transaction upper-bound probability of T_q, that is $p(X, T_q) \leq tubp(T_q)$.*

Proof Since $tubp(T_q) = max\{p(i_1, T_q), p(i_2, T_q), \ldots, p(i_j, T_q)\}$,

$$p(i_j, T_q) \in (0, 1] \Rightarrow p(X, T_q) = \prod_{i_j \in X} p(i_j, T_q) \leq p(i_j, T_q) \leq max\{p(i_j, T_q)\} = tubp(T_q).$$

Therefore, it follows that $p(X, T_q) \leq tubp(T_q)$. □

Definition 10 (Transaction upper-bound weighted probability, ***tubwp***) The transaction upper-bound weighted probability of a transaction T_q is denoted as $tubwp(T_q)$, and defined as $tubwp(T_q) = tubw(T_q) \times tubp(T_q)$.

For example, $tubwp(T_1) = tubw(T_1) \times tubp(T_1) = 0.9 \times 1.0 (= 0.9)$. Moreover, the transaction upper-bound weighted probability of each transaction is shown in Table 7.

Definition 11 (Itemset upper-bound weighted probability, ***iubwp***) The itemset upper-bound weighted probability of an itemset X is defined as:

$$iubwp(X) = \sum\nolimits_{X \subseteq T_q \wedge T_q \in D} tubwp(X, T_q).$$

In the example of Table 1, $iubwp(A) = tubwp(T_1) + tubwp(T_3) + tubwp(T_6) + tubwp(T_{10}) = 0.9 + 0.9 + 0.9 + 0.9 (= 3.6)$ and $iubwp(ACEF) = tubwp(T_3) (= 0.9)$

Definition 12 (High upper-bound expected weighted itemset, HUBEWI) An itemset X in D is called a high upper-bound expected weighted itemset (HUBEWI) if its $iubwp(X)$ is no less than the minimum expected weighted count that is if $iubwp(X) \geq \varepsilon \times |D|$.

Consider the running example and that the minimum expected weighted-support threshold is set to 10%. The itemset (A) is a HUBEWI since $iubwp(A) (= 3.6 > 10\% \times 10)$, while the itemset $(ACEF)$ is not a HUBEWI since $iubwp(ACEF) (= 0.9 < 10\% \times 10)$.

Theorem 4 (HUBEW downward closure property, HUBEWDC property) *Let X^k be a k-itemset and X^{k-1} be a length k-1 subset of X^k. If X is a HUBEWI, any subset of X is also a HUBEWI. Thus, $iubwp(X^k) \leq iubwp(X^{k-1})$.*

Proof Since $X^{k-1} \subseteq X^k$ the set of $TIDs(X^k \subseteq T_q) \subseteq TIDs(X^{k-1} \subseteq T_q)$. Thus:

$$\begin{aligned} iubwp(X^k) &= \sum\nolimits_{X^k \subseteq T_q \wedge T_q \in D} tubwp(T_q) \\ &\leq \sum\nolimits_{X^{k-1} \subseteq T_q \wedge T_q \in D} tubwp(T_q) \\ &= iubwp(X^{k-1}). \end{aligned}$$

Therefore, if X^k is a HUBEWI, any subset X^{k-1} of X^k is also a HUBEWI. □

Corollary 1 *If an itemset X^k is a HUBEWI, every subset X^{k-1} of X^k is also a HUBEWI.*

Corollary 2 *If an itemset X^k is not a HUBEWI, no superset X^{k+1} of X^k is a HUBEWI.*

Theorem 5 ***(HEWIs ⊆ HUBEWIs)*** *The high upper-bound expected weighted downward closure (HUBEWDC) property of HUBEWIs ensures that HEWIs ⊆HUBEWIs in an uncertain database. Thus, if an itemset X is not a HUBEWI, no superset of X is a HEWI.*

Proof Let X be an itemset. Based on definition 3, $w(X) = w(X, T_q)$. From Theorems 2 and 3, it can be obtained that $w(X, T_q) \leq tubw(T_q)$, and $p(X, T_q) \leq tubp(T_q)$.

Table 7 Transaction upper-bound weighted probability of each transaction

TID	1	2	3	4	5	6	7	8	9	10
tubwp	0.9	0.7	0.9	1.0	1.0	0.9	1.0	0.487	1.0	0.9

Thus:

$$
\begin{aligned}
& expWSup(X) = w(X) \times \sum\nolimits_{X \subseteq T_q \wedge T_q \in D} p(X, T_q) \\
&= \sum\nolimits_{X \subseteq T_q \wedge T_q \in D} (w(X) \times p(X, T_q)) \\
&= \sum\nolimits_{X \subseteq T_q \wedge T_q \in D} (w(X, T_q) \times p(X, T_q)) \\
&\leq \sum\nolimits_{X \subseteq T_q \wedge T_q \in D} (tubw(T_q) \times tubp(T_q)) \\
&= \sum\nolimits_{X \subseteq T_q \wedge T_q \in D} tubwp(T_q) \\
&= iubwp(X) \\
& \Rightarrow expWSup(X) \leq iubwp(X).
\end{aligned}
$$

Thus, if an itemset X is not a HUBEWI in an uncertain database D, no superset of X is a HEWI.

The proposed HEWI-Uapriori algorithm will be described next. Since it is based on the designed HUBEWDC property, theorem 5 ensures that HEWI-Uapriori will not consider supersets of a small upper-bound expected weighted itemset (small-UBEWI) as candidate itemsets (**correctness**) and that the complete set of HEWIs can be extracted from the produced HUBEWIs (**completeness**). Therefore, the result of the proposed HEWI-Uapriori algorithm is correct and complete. □

4.2 Proposed HEWI-Uapriori algorithm

The proposed HEWI-Uapriori algorithm mines the high expected weighted itemsets (HEWIs) from an uncertain database using an Apriori-like approach and performs two phases to mine the HEWIs. In the first phase, a set of candidate itemsets HUBEWIs are discovered using a level-wise search approach. Then, in the second phase, the database is scanned again to identify the HEWIs among the HUBEWIs. Based on the aforementioned definitions and theorems, the proposed HEWI-Uapriori algorithm is given below.

As shown in Fig. 1, the proposed HEWI-Uapriori algorithm takes the input as: an uncertain transactional database, D; a weight table, *w-table*; a user-specified minimum expected weighted-support threshold, ε. It first scans the database to find the *tubw*, *tubp* and *tubwp* of each transaction (Lines 1 to 5). The *iubwp* of all 1-itemsets in the uncertain database is calculated (Line 6) to identify HUBEWI^1, the set of HUBEWI of length 1 (Lines 6 to 11). This set is then used to generate C_2, the candidate itemsets of length 2, which are then used for discovering HUBEWI^2. Based on the designed HUBEWDC property, this process is then repeated in a level-wise way until no candidate itemsets can be generated (Lines 13 to 23). A second phase is then performed where an additional database scan is performed to identify the HEWIs among the derived HUBEWIs (Lines 24 to 32), notice that only the *expSup* value of each candidate is needed to calculate from the processed database since the weight of an itemset can be easily obtained from the *w-table* (Lines 25 to 26).

Algorithm 1: HEWI-Uapriori

INPUT: D, an uncertain transactional database;
w-table, a weight table;
ε, a user-specified minimum expected weighted-support threshold.
OUTPUT: The set of high expected weighted itemsets (HEWIs).
/ the first phase, scan D to find HUBEWIs */*
1. **for** each transaction $T_q \in D$ **do**
2. calculate $tubw(T_q) = max\{w(i_1, T_q), w(i_2, T_q), \ldots, w(i_j, T_q)\}$.
3. calculate $tubp(T_q) = max\{p(i_1, T_q), p(i_2, T_q), \ldots, p(i_j, T_q)\}$.
4. calculate $tubwp(T_q) = tubw(T_q) \times tubp(T_q)$.
5. **end for**
6. **for** each item $i_j \in D$ **do**
7. calculate $iubwp(i_j)$
8. **if** $iubwp(i_j) \geq \varepsilon \times |D|$ **then**
9. $HUBEWI^1 \leftarrow HUBEWI^1 \cup i_j$.
10. **end if**
11. **end for**
12. set $k \leftarrow 2$.
13. **while** $HUBEWI^{k-1} \neq$ ***null***, **do**
14. $C_k = Apriori\_gen(HUBEWI^{k-1})$.
15. **for** each k-itemset $X \in C_k$ **do**
16. scan D to calculate $iubwp(X) = \sum_{X \subseteq T_q \wedge T_q \in D} tubwp(T_q)$.
17. **if** $iubwp(X) \geq \varepsilon \times |D|$ **then**
18. $HUBEWI^k \leftarrow HUBEWI^k \cup \{X\}$.
19. **end if**
20. **end for**
21. $k \leftarrow k+1$.
22. **end while**
23. $HUBEWIs \leftarrow \bigcup_k HUBEWI^k$
/ the second phase, scan D to find HEWIs from HUBEWIs */*
24. **for** each k-itemset $X \in HUBEWIs$ **do**
25. scan D to calculate $expSup(X) = \sum_{X \subseteq T_q \wedge T_q \in D} (\prod_{i_j \in X} p(i_j, T_q))$.
26. $expWSup(X) = w(X) \times expSup(X)$.
27. **if** $expWSup(X) \geq \varepsilon \times |D|$ **then**
28. $HEWI^k \leftarrow HEWI^k \cup \{X\}$.
29. **end if**
30. **end for**
31. $HEWIs \leftarrow \bigcup HEWI^k$.
32. **return** *HEWIs*

Fig. 1 Pseudo code of the proposed HEWI-Uapriori algorithm

5 An illustrated example of the HEWI-Uapriori algorithm

In this section, an example is given to illustrate the proposed HEWI-Uapriori algorithm to mine the high expected weighted itemsets (HEWIs) from an uncertain database. Consider the uncertain database shown in Table 1 and the user-specified weight of each item as defined in Table 2.

The minimum expected weighted-support threshold is set to 10 %. Thus, the minimum expected weighted support is calculated as 10 % ×10 (= 1.0).

The database is first scanned to find the transaction upper-bound weight, transaction upper-bound probability, and transaction upper-bound weighted probability of each transaction. The results are respectively shown in Tables 5, 6 and 7. Afterwards the itemset upper-bound weighted probabilities of all 1-itemsets appearing in the uncertain database are calculated. For instance, consider itemset (A) to illustrate this step. This itemset appears in transactions T_1, T_3, T_6, and T_{10}. Thus, $iubwp(A) = tubwp(T_1)+tubwp(T_3)+tubwp(T_6)+tubwp(T_{10}) = 0.9 + 0.9 + 0.9 + 0.9$ (= 3.6 >1.0). As a result, itemset (A) is not a HUBEWI^1. The final calculated set HUBEWI^1 is {$iubwp(A)$:3.6, $iubwp(B)$:6.1875, $iubwp(C)$:5.6, $iubwp(D)$: 4.7, $iubwp(E)$:4.1875, $iubwp(F)$:24.5}. The variable k is then set to 2 to generate C_2 for mining HUBEWI^2 using an Apriori-like approach. The original database is then rescanned to calculate the itemset upper-bound weighted probability ($iubwp$) of itemsets in C_2. The discovered set C_2 is shown in Table 8.

The discovered set C_2 shown in Table 8 is then used to find the set HUBEWI^2. The result is shown in Table 9.

This process is then repeated until no candidate itemsets can be generated. The final set of HUBEWIs is shown in Table 10.

After that, an additional database scan is performed to find the actual HEWIs among HUBEWIs. For example, the expected weighted support of itemset (A) is calculated as $expWSup(A) = w(A) \times expSup(A) = 0.2 \times 2.15$ (= 0.43) which is smaller than 1.0. Thus, the itemset (A) is not a HEWI and is thus discarded. The resulting set of HEWIs is shown in Table 4.

Table 8 Discovered C_2 for mining HUBEWI^2

2-itemset	*iubwp*
(*AB*)	2.7
(*AC*)	3.6
(*AD*)	0
(*AE*)	2.7
(*AF*)	1.8
(*BC*)	4.7
(*BD*)	3.0
(*BE*)	3.2875
(*BF*)	2.8
(*CD*)	2.0
(*CE*)	3.7
(*CF*)	1.8
(*DE*)	1.0
(*DF*)	2.7
(*EF*)	0.9

Table 9 Discovered HUBEWI^2

2-itemset	*iubwp*
(*AB*)	2.7
(*AC*)	3.6
(*AE*)	2.7
(*AF*)	1.8
(*BC*)	4.7
(*BD*)	3.0
(*BE*)	3.2875
(*BF*)	2.8
(*CD*)	2.0
(*CE*)	3.7
(*CF*)	1.8
(*DE*)	1.0
(*DF*)	2.7

6 Experimental results

In this section we present extensive experiments to evaluate the performance of the proposed HEWI-Uapriori algorithm on both synthetic and real-life datasets. Because the proposed high expected weighted itemset (HEWI) mining problem is a novel problem that consider both the weight

Table 10 Final discovered HUBEWIs

k-itemset	*iubwp*	*k*-itemset	*iubwp*
(*A*)	3.6	(*DE*)	1.0
(*B*)	6.1875	(*DF*)	2.7
(*C*)	5.6	(*ABC*)	2.7
(*D*)	4.7	(*ABE*)	1.8
(*E*)	4.1875	(*ABF*)	1.8
(*F*)	4.5	(*ACE*)	2.7
(*AB*)	2.7	(*ACF*)	1.8
(*AC*)	3.6	(*BCD*)	2.0
(*AE*)	2.7	(*BCE*)	2.8
(*AF*)	1.8	(*BCF*)	1.8
(*BC*)	4.7	(*BDE*)	1.0
(*BD*)	3.0	(*BDF*)	1.0
(*BE*)	3.2875	(*CDF*)	1.0
(*BF*)	2.8	(*ABCE*)	1.8
(*CD*)	2.0	(*ABCF*)	1.8
(*CE*)	3.7	(*BCDF*)	1.0
(*CF*)	1.8		

Table 11 Dataset parameters

#|*D*|	Total number of transactions
AvgLen	Average transaction length
#|*I*|	Number of distinct items
Type	Dataset type (sparse or dense)

and existential probability constraint, the existing Uapriori algorithm [10] (for mining expected support frequent itemsets in an uncertain dataset) and the existing PWA algorithm [13] (for mining the weighted frequent itemsets in a precise dataset) are used as benchmark algorithms for comparison with the proposed HEWI-Uapriori algorithm.

6.1 Experimental setup and datasets description

All algorithms used in the experiments are implemented in Java. Experiments were performed on a computer having an Intel Core2 Duo 2.8 GHz processor with 4GB of main memory, running the 32 bit Microsoft Windows 7 operating system. Both real-life [1] and synthetic datasets [6] were used in the experiments. A simulation model was developed to generate weights for all items in the dataset in the (0,1] interval to fill the weight table. In addition, the existential probabilities of items in transactions have been randomly generated in the (0, 1] interval. The parameters of the datasets are presented in Table 11, and the characteristics are depicted in Table 12.

Experiments were conducted to show the performance in terms of execution time, memory consumption, number of discovered patterns for the weighted frequent itemsets (WFIs), expected support frequent itemsets (EFIs), and the designed high expected weighted itemsets (HEWIs). A scalability experiment was also carried out to show the effectiveness and efficiency of the proposed algorithm.

6.2 Performance of runtime

In this section, the runtime of the proposed HEWI-Uapriori and other compared algorithms are first compared. The size of the test datasets is fixed and the user-specified minimum expected weighted-support threshold *minEWSup* is changed. Note that here *minEWSup* refers to the minimum expected support threshold (*minESup*) as used in the Uapriori algorithm, the minimum weighted-support threshold (*minWSup*) as used by the PWA algorithm, and the minimum expected weighted-support threshold (ε) as used in the proposed HEWI-Uapriori algorithm. The runtime of the HEWI-Uapriori and other compared algorithms under various minimum expected weighted-support thresholds for the different datasets are shown in Fig. 2 Note that runtime includes both computation time and I/O time.

Table 12 Dataset characteristics

Dataset	#|*D*|	AvgLen	#|*I*|	Type
foodmart	21,556	4.4	1,559	Sparse
retail	88,162	10.3	16,470	Sparse
mushroom	8,124	23	119	Dense
T10I4D100K	100,000	10.1	870	Sparse

From Fig. 2, it can be observed that the proposed HEWI-Uapriori algorithm is slower than the Uapriori algorithm but that it is faster than the projection-based PWA algorithm. It can also be observed that HEWI-Uapriori is faster than the traditional weighted frequent itemset mining algorithm PWA for the retail and T10I4D100K datasets, but is slower than PWA for the foodmart and mushroom datasets under various *minEWSup* values.

For example, Fig. 2d shows the runtime of the three algorithms for the T10I4D100K dataset when the *minEWSup* threshold is varied from 0.01 % to 0.5 %, with a 0.1 % increment. The runtime of HEWI-Uapriori is much faster than PWA, but worse than Uapriori when *minEWSup* is set lower. The reason is that when the *minEWSup* threshold is set relatively low, the two-phase HEWI-Uapriori algorithm has to maintain a large amount of HUBEWIs in memory before identifying the HEWIs, which is a little time-consuming. When *minEWSup* is set high, the upper-bound is close to the actual expected weighted-support value. Thus, fewer redundant upper-bound patterns are generated, and so the performance of the HEWI-Uapriori algorithm is close to the Uapriori algorithm. In general, when more constraints are considered, fewer patterns are produced.

A very interesting result is that the performance of the projection-based PWA algorithm is better than the HEWI-Uapriori algorithm for the foodmart dataset. The reason is that for a very sparse dataset, such as foodmart (average transaction length of 4.4 items), many unpromising candidates can be pruned by the projection model and the weighted upper-bound. However, this strategy is not very efficient for dense datasets or even for normal datasets. In these cases, the PWA algorithm requires more time to perform dataset projection in mining process. The reason is that for dense datasets such as mushroom (average transaction length of 23 items), few itemsets are pruned. Almost all of them are used to perform projections to generate candidate datasets that are explored recursively. Thus, both the projection model and the weighted upper-bound strategy cannot play a positive role in pruning candidates in dense datasets.

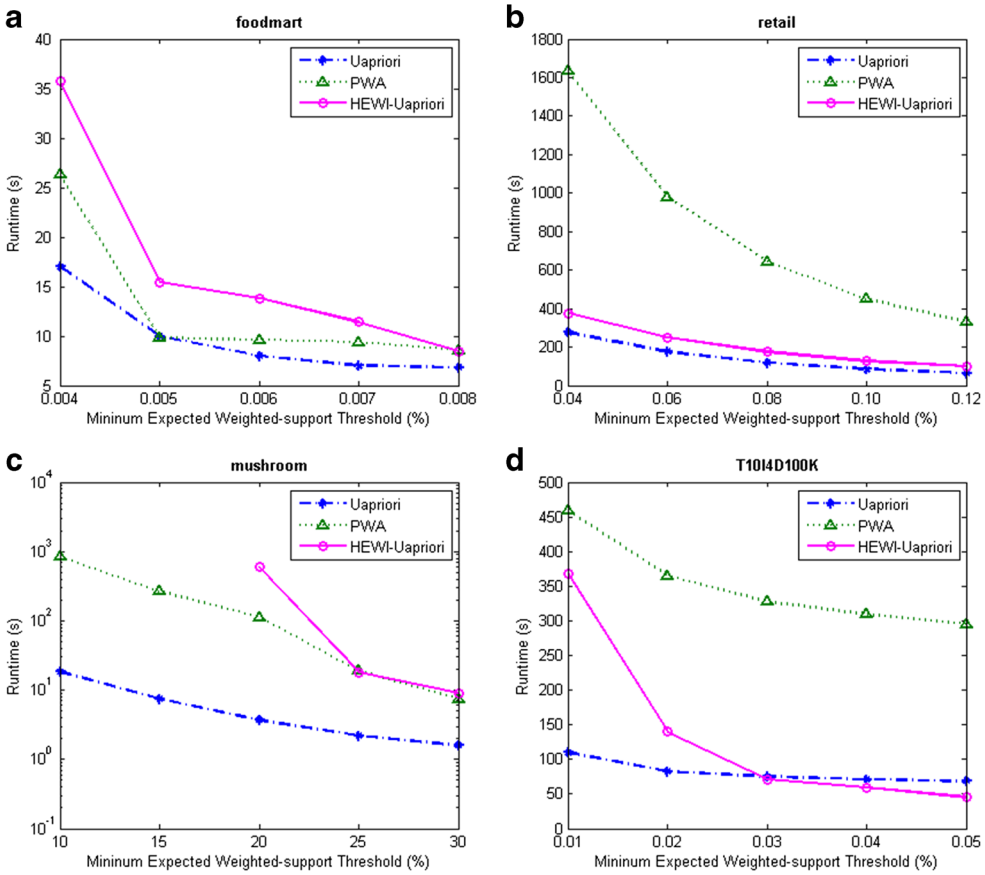

Fig. 2 Runtime of compared algorithms under various minimum expected weighted-support thresholds

6.3 Patterns analysis

The number of EFIs, WFIs and HEWIs patterns are compared in this section. Notice that the Uapriori algorithm generates EFIs, the PWA algorithm generates WFIs, and that the proposed HEWI-Uapriori algorithm discovers HEWIs. Those three different kinds of patterns (EFIs, WFIs and HEWIs), are compared under various *minEWSup* threshold values. Results are shown in Fig. 3. Then, the number of k-itemsets is compared for a fixed *minEWSup* value. Finally, the compression ratio of the HEWFIs which are generated by the proposed HEWI-Uapriori algorithm is also evaluated.

From Fig. 3, it can be observed that whether in sparse or dense datasets, the number of EFIs is always smaller than the number of WFIs, but is larger than the number of HEWIs under various minimum expected weighted-support thresholds. On one hand, it indicates that numerous WFIs are discovered but few WFIs are HEWIs when the weight factor of each item is considered. On the other hand, numerous EFIs are discovered because of their high expected frequency but they do not have a high weight. Hence, those patterns (EFIs and WFIs) may not be helpful in decision-making (e.g. for a manager or retailer) and may be uninteresting for real-world applications. Thus, the discovered HEWIs can be considered as valuable patterns compared to the EFIs and WFIs found by traditional approaches since both the dataset uncertainty and the weight factor occur in real-life situations.

The above observations generally hold when the minimum expected weighted-support threshold is set to a small value. When the *minEWSup* threshold is raised, fewer HEWIs are produced by the proposed algorithm compared to the traditional frequent itemset mining algorithms for discovering EFIs in uncertain datasets, as well as those for mining WFIs. This is because the proposed algorithm discovers HEWIs by considering the high expected weighted-frequent constraint, which considers both the probability and weight measures. The Uapriori and PWA algorithms, however, are used to discover the EFIs and WFIs. The Uapriori algorithm only considers the expected support constraint but it does not consider the weight factor. The PWA algorithm only uses the expected weighted support constraint to find interesting patterns. From these results, it can found that the number of EFIs WFIs and HEWIs all dramatically decreased when *minEWSup* is increased. The reason is that for a high *minEWSup* value, numerous unpromising candidates can be pruned early, and thus algorithms can avoid keeping them or considering them after in the mining process. Moreover, fewer interesting patterns satisfy the higher minimum condition value when *minEWSup* is increased.

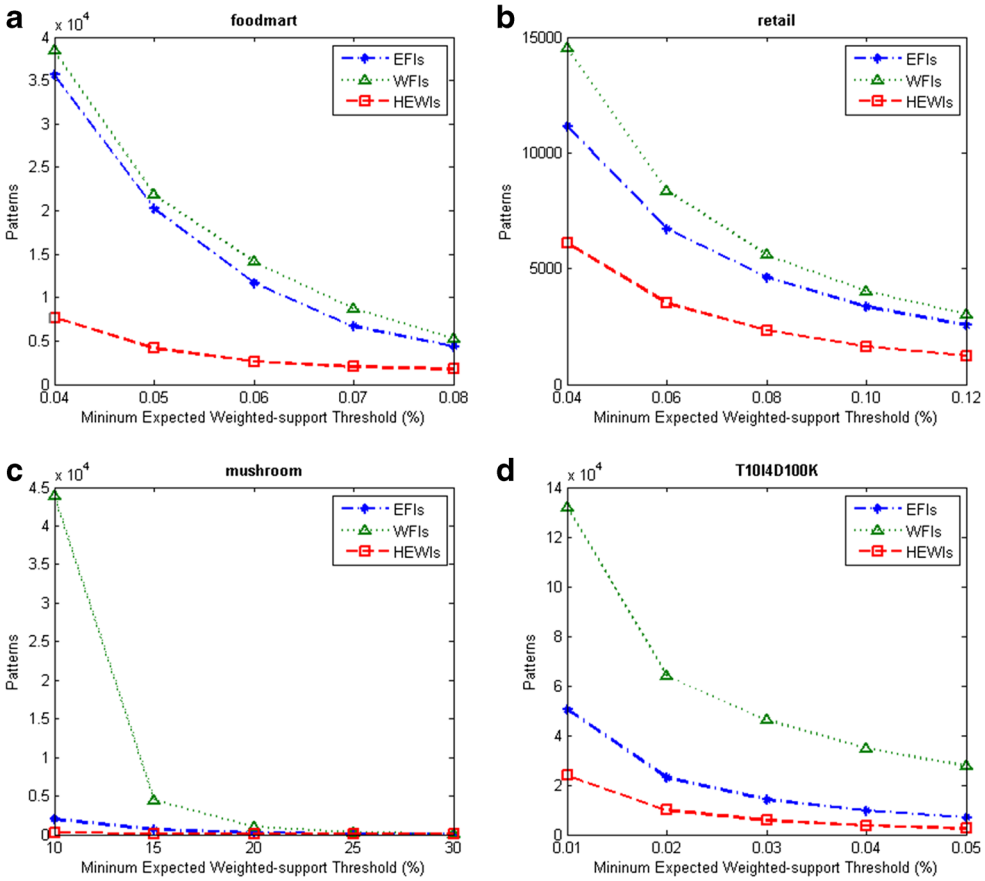

Fig. 3 Number of EFIs, WFIs and HEWIs under various minimum expected weighted-support thresholds

The number of k-itemsets discovered by the compared algorithms is also evaluated. Results are shown in Fig. 4. Here, a selected "middle" threshold of *minEWSup* (*minESup* *minWSup* or ε) in each subpicture of Fig. 4 is used.

In Fig. 4, it is obvious that the number of proposed HEWIs patterns is generally less than or equal to both the number of EFIs and WFIs patterns. Fewer and more meaningful patterns are thus discovered by the proposed algorithm. For WFIs patterns (as shown in Fig. 4), although fewer 1-itemsets are found, more k-itemsets ($k \geq 2$) are output. According to Fig. 4c and d, the number of 1-itemsets that are EFIs is close to the number of WFIs, but the k-itemsets ($k \geq 2$) are much fewer than WFIs. This is reasonable because the downward closure property of the final derived patterns, either WFIs or HEWIs, does not hold in the PWA algorithm and the proposed algorithm. In the PWA algorithm, an itemset that is a non-weighted frequent itemset may have supersets that are weighted frequent itemsets. In the proposed HEWI-Uapriori algorithm, when considering both the weight and probability properties, fewer HEWIs are produced. This is because as larger itemsets are explored, the probability that these k-itemsets occur becomes smaller. Thus, the HEWIs are not only much fewer than the weighted or expected support patterns, but are also more precise than the other patterns.

Furthermore, the compression ratio of high expected weightedfrequent itemsets (abbreviated as *RateHEWIs*) is investigated in the following. This ratio is defined as follows.

$$RateHEWIs^1 = \frac{|EFIs - HEWIs|}{|EFIs|}$$

$$RateHEWIs^2 = \frac{|WFIs - HEWIs|}{|WFIs|}$$

Results are shown in Table 13.

From Table 13, it can be observed that the compression ratio achieved by mining HEWIs instead of EFIs or WFIs is very high, and $RateHEWIs^1$ is larger than $RateHEWIs^2$ for the same minimum threshold. This means that numerous redundant and meaningless patterns are effectively eliminated. In other words, neither the high expected frequent itemsets nor the weighted frequent patterns are able to express the importance of patterns well. As more constraints are applied in patterns mining, more meaningful results and fewer patterns are discovered, which indicates that the proposed novel methodology is reasonable and acceptable.

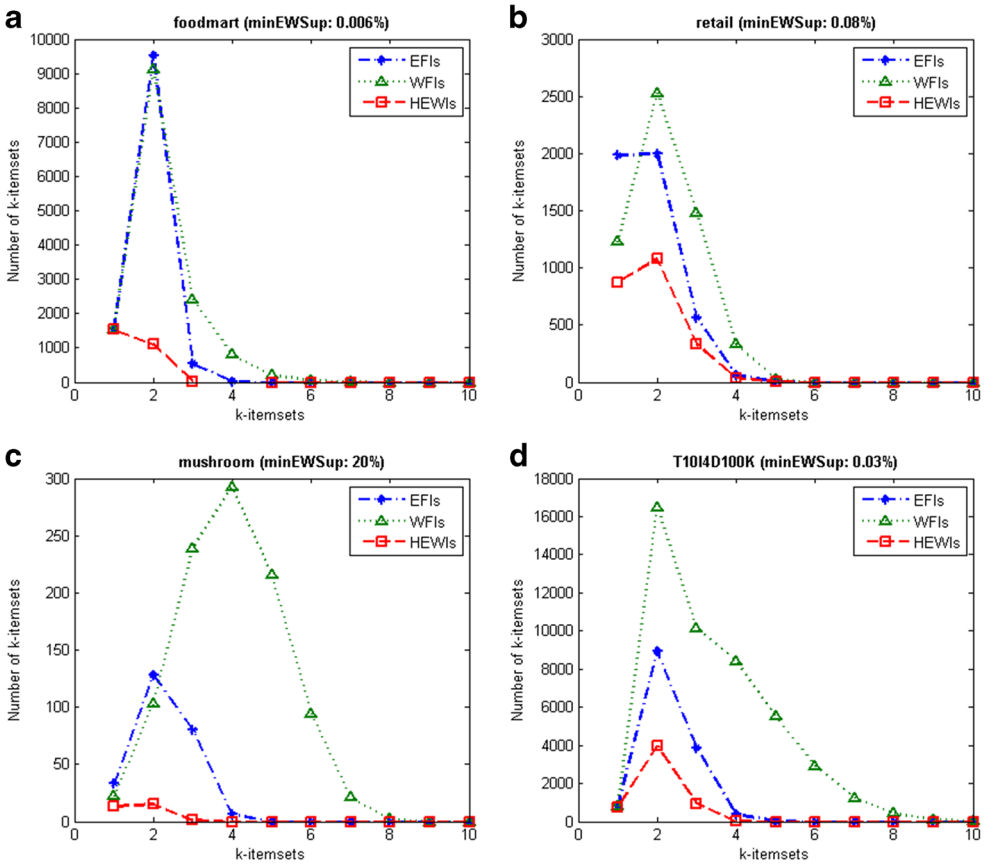

Fig. 4 Number of *k*-itemsets under various minimum expected weighted-support thresholds

Table 13 Analysis of *RateHEWIs* under various minimum expected weighted-support thresholds

Foodmart	0.004 %	0.005 %	0.006 %	0.007 %	0.008 %
EFIs	35624	20211	11647	6697	4362
WFIs	38411	21808	14118	8735	5262
HEWIs	7696	4141	2649	2005	1749
RateHEWIs^1	78 %	79 %	77 %	70 %	60 %
RateHEWIs^2	80 %	81 %	81 %	77 %	66 %
retail	0.04 %	0.06 %	0.08 %	0.10 %	0.12 %
EFIs	11141	6707	4610	3354	2551
WFIs	14522	8346	5591	4030	3038
HEWIs	6101	3501	2326	1630	1240
RateHEWIs^1	45 %	48 %	49 %	51 %	51 %
RateHEWIs^2	58 %	58 %	58 %	59 %	59 %
mushroom	10 %	15 %	20 %	25 %	30 %
EFIs	2001	610	247	114	70
WFIs	43833	4472	990	272	101
HEWIs	297	85	29	15	7
RateHEWIs^1	85 %	96 %	88 %	87 %	90 %
RateHEWIs^2	99 %	98 %	97 %	94 %	93 %
T10I4D100K	0.01 %	0.02 %	0.03 %	0.04 %	0.05 %
EFIs	50357	22928	14020	9559	6920
WFIs	132040	63945	46102	34820	27547
HEWIs	23905	9927	5674	3565	2482
RateHEWIs^1	52 %	56 %	59 %	63 %	64 %
RateHEWIs^2	82 %	84 %	87 %	90 %	91 %

6.4 Performance of memory consumption

Additional experiments were performed to assess the memory consumption of the proposed algorithm. The experiments were conducted using the parameters described in Section 6.2. Results are shown in Fig. 5.

In Fig. 5, it can be seen that the proposed HEWI-Uapriori algorithm always requires more memory than the compared algorithms for the four datasets. In particular, the PWA algorithm requires nearly constant memory under various *minEWSup* for the four datasets. The HEWI-Uapriori algorithm requires the most memory among the three compared algorithms. There are several reasons for this. First, Uapriori only considers the expected support of patterns, and uses the downward closure property to directly prune numerous unpromising candidates of EFIs. Thus, it generates fewer candidates than HEWI-Uapriori. HEWI-Uapriori, however, needs to calculate the expected weighted support of the patterns to determine whether a pattern is a candidate or not. Because the downward closure property does not hold for mining HEWIs, HEWI-Uapriori applies the proposed *iubwp* upper-bound strategy to prune numerous candidates of HEWIs without missing any HEWIs. This means that after the first dataset scan, more relevant information and candidates from the dataset are kept in main memory in each level of the level-wise search. Hence, the HEWI-Uapriori algorithm consumes the most memory. For example, for the retail dataset, the HEWI-Uapriori algorithm requires 175 MB of memory on average, the Uapriori consumes 130 MB of memory on average, while PWA only needs 8 MB of memory on average, as it can be observed in Fig. 5b.

It can also be easily seen that the performance gap between the HEWI-Uapriori and other compared algorithms becomes smaller as *minEWSup* increases. For instance, when *minEWSup* is set to 0.004 %, the HEWI-Uapriori, Uapriori, and the PWA algorithms respectively consume 49 MB, 25 MB and 14 MB of memory for the foodmart dataset (see Fig. 5a). When *minEWSup* is set to 0.008 %, the three compared algorithms respectively need 23 MB, 18 MB and 10MB of memory (see Fig. 5a). The reason for this reduction is that unpromising candidates can be pruned earlier, which reduces the size of the search space, and in the case of HEWI-Uapriori fewer candidates are maintained in memory during mining. Moreover, as the upper-bound of the HEWI-Uapriori algorithm becomes closer to the actual expected weighted-support value, fewer redundant upper-bound patterns are generated. Therefore, the

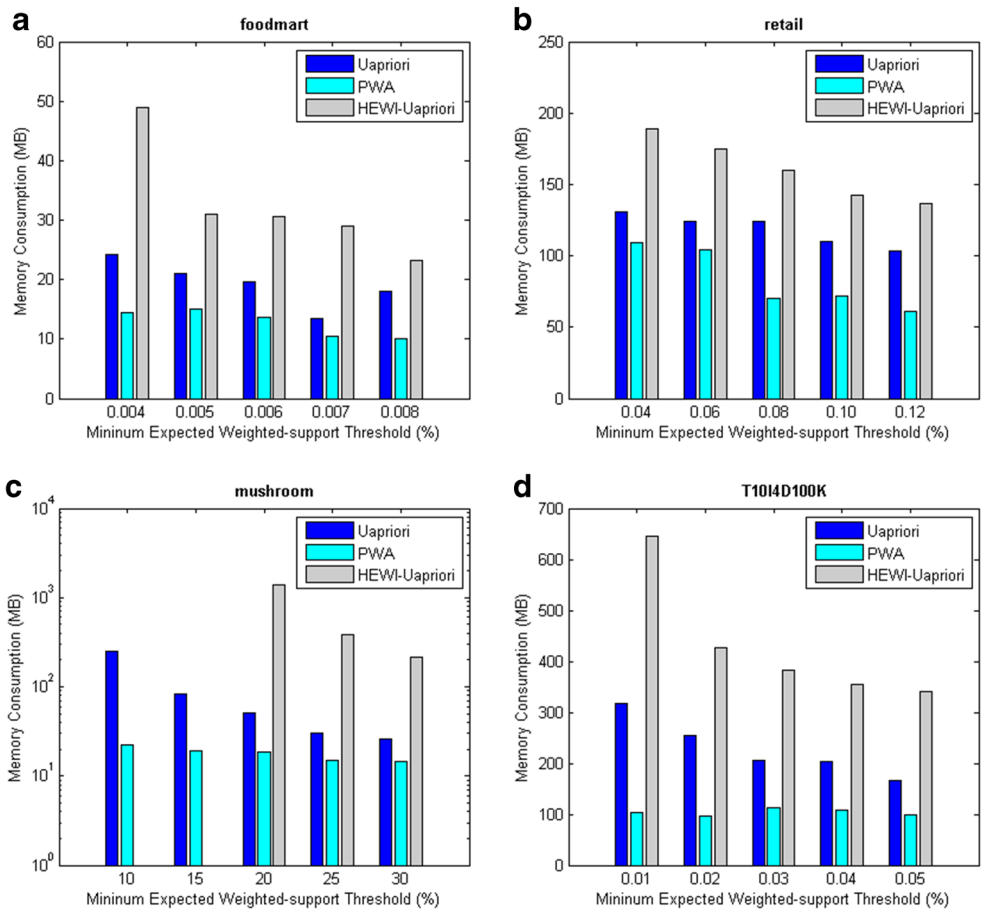

Fig. 5 Memory consumption under various minimum expected weighted-support thresholds

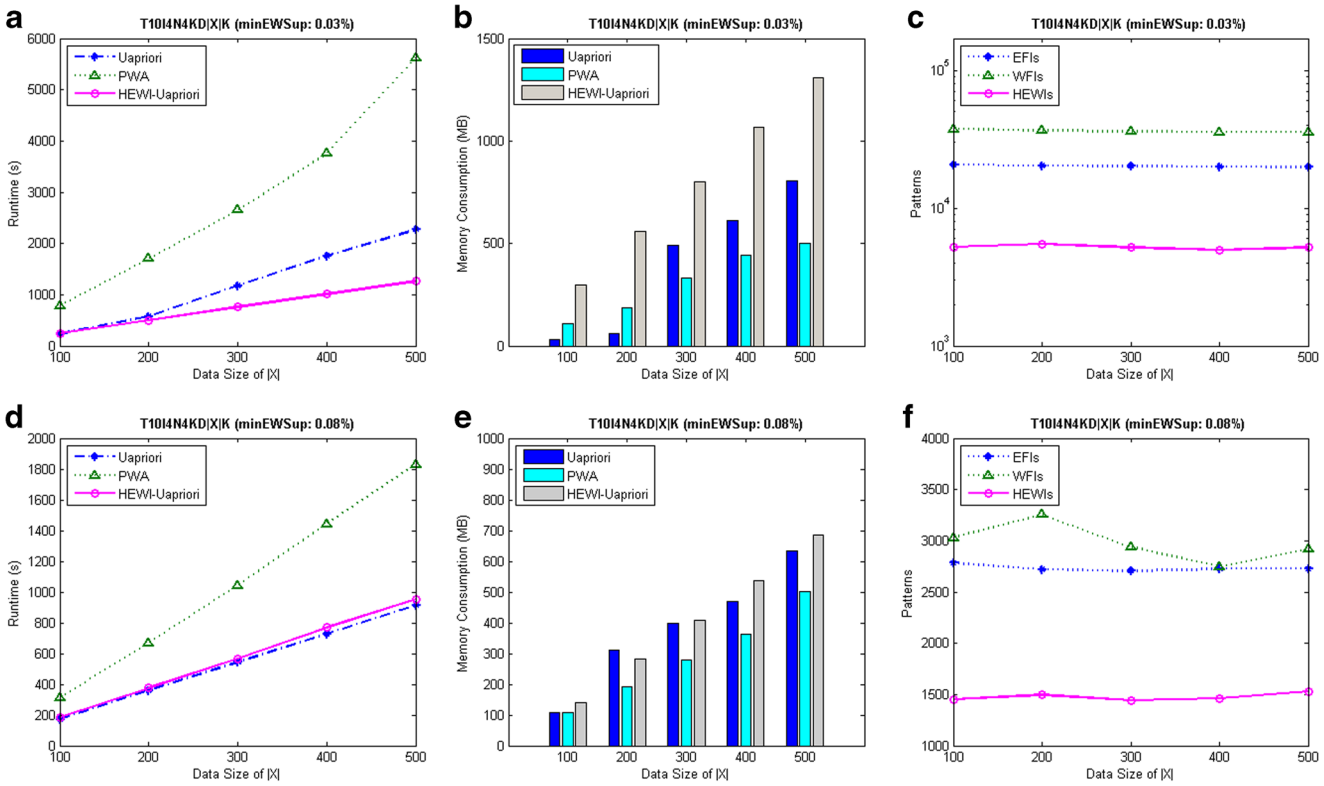

Fig. 6 Scalability of the compared algorithms in different dataset size

memory usage performance of the HEWI-Uapriori algorithm becomes close to that of the compared algorithms when *minEWSup* is set higher.

6.5 Scalability experiment

The scalability of the proposed HEWI-Uapriori and other compared algorithms is also evaluated. The experiments are performed on a set of synthetic datasets named T10I4N4KD|X|K where |X| is varied. Figure 6a to f respectively show the results in terms of runtime, pattern count and memory consumption when *minEWSup* was varied from 0.03 % to 0.08 %, respectively.

It can be seen in Fig. 6 that all the compared algorithms have good scalability in terms of runtime, and that the HEWI-Uapriori algorithm perform better than the two others. As shown in Fig. 6a, when *minEWSup* is set to 0.03%, and as expected, the HEWI-Uapriori algorithm is faster than Uapriori and PWA when the dataset size varies from 100K to 500K transactions. The reasons are as follows. The HEWI-Uapriori uses a level-wise approach that generates candidates, and all candidates need to be tested to filter those that are unpromising. It adopts an efficient pruning strategy namely HUBEWDC property to prune unpromising k-itemsets early. As an Apriori-like approach, it still need to repeatedly scanning the dataset, so it may perform worse when the *minEWSup* was set lower, as shown in 6(d). Furthermore, when the value of |X| is increased both the transaction length and the probability that larger itemsets appear in the transaction dataset increases. Thus, for all algorithms, more patterns needs to be considered to discover the interesting patterns. As a result, execution times increase significantly. In particular, these results show that the HEWI-Uapriori algorithm is much more scalable than the other algorithms. The larger the dataset size is, the bigger the difference between the compared algorithms in terms of runtime and memory consumption. The above mentioned reasons can be used to explains results about the number of patterns depicted in Fig. 6c and f. Thus, the proposed HEWI-Uapriori algorithm can achieve good scalability, and in general have a behavior that is acceptable for real-world applications.

7 Conclusion and future work

The problems of weighted frequent itemset mining and expected support frequent itemset mining have been extensively studied to respectively mine patterns based on the weight constraint and the existential probability constraint.

Mining the weighted frequent itemsets in an uncertain database had, however, not yet been proposed. In this paper, a novel problem called high expected weighted itemset (HWEI) mining is proposed to consider both the weight and existential probability constraints. The proposed HEWIs can thus be used to reveal more interesting and meaningful information, especially when the importance of each item is considered in an uncertain database.

The developed algorithm is named HEWI-Uapriori. Based on the proposed upper-bound of HEWI and downward closure property, HEWI-Uapriori adopts an Apriori-like approach, which is performed in two phases. In the first phase, it mines the high upper-bound expected weighted itemsets (HUBEWIs) in a level-wise way. In the second phase, it identifies actual HEWIs among HUBEWIs by performing an additional database scan. By using the proposed high upper-bound expected weighted downward closure (HUBEWDC) property, the search space for mining the HEWIs can be greatly reduced. Both real-life and synthetic datasets have been used to compare the performance of the proposed algorithm with traditional algorithms for weighted frequent itemset mining and expected support frequent itemset mining. Results show that the proposed algorithm has better performance and scalability.

Because in real-life, a database may dynamically change as the result of insertion, deletion and modification operations, for future work, we plan to extend the proposed algorithm to add the capability of incrementally mining HEWIs. Another interesting possibility for future work is to design algorithms for mining HEWIs using more compressed and condensed tree structures.

Acknowledgments This research was partially supported by the Tencent Project under grant CCF-TencentRAGR20140114, by the Shenzhen Peacock Project, China, under grant KQC201109020055A, by the Natural Scientific Research Innovation Foundation in Harbin Institute of Technology under grant HIT.NSRIF.2014100, and by the Shenzhen Strategic Emerging Industries Program under grant ZDSY20120613125016389.

References

1. (2012) Frequent itemset mining dataset repository. Available: http://fimi.ua.ac.be/data/
2. Aggarwal CC, Li Y, Wang J, Wang J (2009) Frequent pattern mining with uncertain data, ACM SIGKDD international conference on knowledge discovery and data mining, pp 29–38
3. Aggarwal CC, Yu PS (2009) A survey of uncertain data algorithms and applications. IEEE Trans Knowl Data Eng 21(5):609–623
4. Aggarwal CC (2010) Managing and mining uncertain data, managing and mining uncertain data
5. Agrawal R, Srikant R (1994) Fast algorithms for mining association rules in large databases, the international conference on very large data bases, pp 487–499
6. Agrawal R, Srikant R (1994) Quest synthetic data generator. Available: http://www.Almaden.ibm.com/cs/quest/syndata.html
7. Bernecker T, Kriegel HP, Renz M, Verhein F, Zuefl A (2009) Probabilistic frequent itemset mining in uncertain databases, ACM SIGKDD international conference on knowledge discovery and data mining, pp 119–128
8. Cai CH, Fu AWC, Cheng CH, Kwong WW (1998) Mining association rules with weighted items, the international database engineering and applications symposium, pp 68–77
9. Chen MS, Han J, Yu PS (1996) Data mining: An overview from a database perspective. IEEE Trans Knowl Data Eng 8(6):866–883
10. Chui CK, Kao B, Hung E (2007) Mining frequent itemsets from uncertain data, advances in knowledge discovery and data mining
11. Geng L, Hamilton HJ Interestingness measures for data mining: A survey. ACM Comput Surv 38(3):2006
12. Han J, Pei J, Yin Y, Mao R (2004) Mining frequent patterns without candidate generation: A frequent-pattern tree approach. Data Min Knowl Disc 8(1):53–87
13. Lan GC, Hong TP, Lee HY, Lin CW (2013) Mining weighted frequent itemsets, the workshop on combinatorial mathematics and computation theory, pp 85–89
14. Lan GC, Hong TP, Lee HY (2014) An efficient approach for finding weighted sequential patterns from sequence databases. Appl Intell 41(2):439–452
15. Leung CKS, Mateo MAF, Brajczuk DA (2008) A tree-based approach for frequent pattern mining from uncertain data, advances in knowledge discovery and data mining, pp 653–661
16. Leung CKS, Hao B (2009) Mining of frequent itemsets from streams of uncertain data, IEEE international conference on data engineering, pp 1663–1670
17. Lin CW, Hong TP, Lu WH (2009) The Pre-FUFP algorithm for incremental mining. Expert Syst Appl 36(5):9498–9505
18. Lin CW, Hong TP (2011) Temporal data mining with up-to-date pattern trees. Expert Syst Appl 38(12):15143–15150
19. Lin CW, Hong TP (2012) A new mining approach for uncertain databases using cufp trees. Expert Syst Appl 39(4):4084–4093
20. Pei J, Han J, Mortazavi-Asl B, Wang J, Pinto H, Chen Q, et al. (2004) Mining sequential patterns by pattern-growth: The prefixspan approach. IEEE Trans Knowl Data Eng 16(11):1424–1440
21. Srikant R, Agrawal R (1996) Mining sequential patterns: Generalizations and performance improvements, the international conference on extending database technology: advances in database technology, pp 3–17
22. Sun L, Cheng R, Cheung DW, Cheng J (2010) Mining uncertain data with probabilistic guarantees, the 16th ACM SIGKDD international conference on knowledge discovery and data mining, pp 273–282
23. Tang P, Peterson EA (2011) Mining probabilistic frequent closed itemsets in uncertain databases. The Annual Southeast Regional Conference, pp 86–91
24. Tao F, Murtagh F, Farid M (2003) Weighted association rule mining using weighted support and significance framework, ACM SIGKDD international conference on knowledge discovery and data mining, pp 661–666
25. Tong Y, Chen L, Cheng Y, Yu PS (2012) Mining frequent itemsets over uncertain databases. The VLDB Endowment 5(11):1650–1661
26. Vo B, Coenen F, Le B (2013) A new method for mining frequent weighted itemsets based on wit-trees. Expert Syst Appl 40(4):1256–1264
27. Wang W, Yang J, Yu PS (2000) Efficient mining of weighted association rules (war), ACM SIGKDD international conference on knowledge discovery and data mining, pp 270–274

28. Yun U, Leggett J (2005) WFIM: Weighted frequent itemset mining with a weight range and a minimum weight, SIAM international conference on data mining, pp 636–640
29. Yun U, Leggett J (2006) WSpan: Weighted sequential pattern mining in large sequential database, IEEE international conference on intelligent systems, pp 512–517
30. Yun U (2008) A new framework for detecting weighted sequential patterns in large sequence databases. Knowl-Based Syst 21(2):110–122
31. Zaki M, Hsiao C (2002) CHARM: An efficient algorithm for closed itemset mining. SIAM Int Conf Data Min 2:457–473

Jerry Chun-Wei Lin received Ph.D. degree in Dept. of Computer Science and Information Engineering in 2010 from National Cheng Kung University, Tainan, Taiwan. He is currently working as an assistant professor at School of Computer Science and Technology, Harbin Institute of Technology Shenzhen Graduate School, China. He has published around 100 research papers in referred journals and international conferences. His interests include data mining, soft computing, privacy preserving data mining and security, social network and cloud computing.

Wensheng Gan is a master student studying at School of Computer Science and Technology, Harbin Institute of Technology Shenzhen Graduate School, China. His research interests include data mining and cloud computing.

Philippe Fournier-Viger is an assistant-professor at University of Moncton, Canada. He received a Ph.D. in Cognitive Computer Science at the University of Quebec in Montreal (2010). He has published more than 75 research papers in refereed international conferences and journals. He is the founder of the popular SPMF open-source data mining library.

Tzung-Pei Hong received his B.S. degree in chemical engineering from National Taiwan University in 1985, and his Ph.D. degree in computer science and information engineering from National Chiao-Tung University in 1992. He is currently a distinguished professor at the Department of Computer Science and Information Engineering and at the Department of Electrical Engineering. He has published more than 400 research papers in international/national journals and conferences and has planned more than fifty information systems. He is also the board member of more than forty journals and the program committee member of more than three hundred conferences. His current research interests include knowledge engineering, data mining, soft computing, management information systems, and www applications. Dr. Hong is a member of the Association for Computing Machinery, the IEEE, the Chinese Fuzzy Systems Association, the Taiwanese Association for Artificial Intelligence, and the Institute of Information and Computing Machinery.

Vincent S. Tseng holds the position of Professor at Department of Computer Science at National Chiao Tung University (NCTU), Taiwan. Currently he also serves as the chair for IEEE CIS Tainan Chapter. Before Dr. Tseng joined NCKU in 1999, he was a postdoctoral research fellow in Computer Science Division of University of California at Berkeley, U.S.A. He served as the president of Taiwanese Association for Artificial Intelligence during 2011-2012 and acted as the director for Institute of Medical Informatics of NCKU during August 2008 and July 2011. During February 2004 and July 2007, he had also served as the director for Informatics Center in National Cheng Kung University Hospital. Dr. Tseng received his Ph.D. degree with major in computer science from National Chiao Tung University, Taiwan, in 1997. He has a wide variety of research interests covering data mining, biomedical informatics, multimedia databases, mobile and Web technologies. He has published around 300 research papers in referred journals and international conferences as well as 15 patents held. He has been on the editorial board of a number of journals including IEEE Transactions on Knowledge and Data Engineering, ACM Transactions on Knowledge Discovery from Data, IEEE Journal of Biomedical and Health Informatics, International Journal of Data Mining and Bioinformatics, Journal of Information Science and Engineering, and Taiwanese Journal of Medical Informatics (associate editor-in-chief). He has also served as chairs/program committee members for a number of premier international conferences related to data mining and biomedical informatics, including KDD, ICDM, SDM, BIBM, BCB, PAKDD, CIKM, IJCAI, etc. He was also the program committee chair for PAKDD2014.