EHUSM: Mining High Utility Sequences with a Pessimistic Utility Model

Tin Truong
Department of Mathematics and Computer Science
University of Dalat
Dalat, Vietnam
tintc@dlu.edu.vn

Anh Tran
Department of Mathematics and Computer Science
University of Dalat
Dalat, Vietnam
anhtn@dlu.edu.vn

Hai Duong
Department of Mathematics and Computer Science
University of Dalat
Dalat, Vietnam
haidv@dlu.edu.vn

Bac Le
VNU - HCMC
University of Science
Ho Chi Minh City, Vietnam
lhbac@fit.hcmus.edu.vn

Philippe Fournier-Viger
School of Natural Sciences and Humanities
Harbin Institute of Technology, Shenzhen
HIT Campus Shenzhen University Town Xili, Shenzhen, China
philfv@hitsz.edu.cn
Abstract. The problem of mining all high utility sequences (HUS) in a quantitative sequence database (QSDB) has attracted the attention of many researchers. It has many applications such as identifying highly profitable purchase patterns of customers in retail stores. However, a sequence (pattern) may appear multiple times in the same input sequence. To handle this situation, most studies adopt an optimistic perspective as they measure the utility of a sequence as the maximum utility among its occurrences. As a result, the utility of a pattern may seem higher than it actually is. Hence, taking business decisions on the basis of a pattern found by traditional high utility sequence mining algorithms may be risky. To address this issue, this paper proposes to adopt a pessimistic model for measuring the utility of patterns, using a novel measure called the minimum utility measure ($u_{\min}$). Although discovering patterns using $u_{\min}$ is desirable to reduce risks for decision-making, designing an efficient algorithm to find patterns using this measure is challenging. The reason is that $u_{\min}$ does not satisfy the downward closure property (DCP) generally used to reduce the search space in sequence mining. To overcome this challenge, two novel upper bounds on $u_{\min}$ are proposed, which are tighter than the well-known sequence-weighted utility upper-bound, and satisfy anti-monotone-like properties that can be weaker than the DCP. Based on these properties, two novel depth and width pruning strategies are designed to eliminate low utility candidate sequences from the search space early. These theoretical results are the basis of a novel algorithm named EHUSM (Efficient High Utility Sequence Mining) for efficiently mining all high utility sequences using $u_{\min}$. Experimental studies on both real-life and synthetic QSDBs show that the proposed algorithm is efficient in terms of runtime and memory.

Keywords: Quantitative sequence database, minimum utility measure, high utility sequence, upper bound, pruning strategy, pessimistic model.

1. Introduction. The problem of frequent sequence mining (FSM) in sequential databases consists of discovering sequences that appear frequently in a set of sequences. It extends the problem of frequent itemset mining in traditional transaction databases by considering the sequential ordering of itemsets. Although FSM is popular, it is unsuitable for many real-life applications, since patterns (sequences) are discovered on the basis of their occurrence frequency rather than their utility (importance). Thus, FSM can discover numerous frequent sequences that are unimportant to the user and miss many important but infrequent sequences. To overcome this limitation of FSM, the problem of high utility sequence mining (HUSM) in quantitative sequence databases was introduced, where each item is associated with a quantity (e.g. a purchase quantity) and a quality that indicates its relative importance (e.g. its unit profit). For example, in market basket analysis, HUSM consists of discovering sequences representing highly profitable sequences of purchases (called high utility sequences), which can be used for marketing purposes. HUSM also has many other applications such as analyzing web logs [1], data from mobile commerce environments [11], gene regulation [20], and healthcare activity-cost event logs [4]. HUSM is a generalization of FSM, and it is more challenging than FSM due to the additional consideration of utility information in sequences and the fact that the utility measure is not monotonic nor anti-monotonic, which makes it unsuitable to reduce the search space [16]. For these reasons, in the last decade, many researchers have studied this problem [2, 3, 9, 10, 12, 21] as well as its extensions [5, 7, 17, 14, 15, 16, 18, 19, 22].

Related work. In previous work related to HUSM, authors have mostly calculated the utility of a sequence using the maximum utility measure $u_{\max}$. The $u_{\max}$ value of a sequence $\alpha$ in each input quantitative sequence $\psi$ is calculated as the maximum utility among those of all occurrences of $\alpha$ in $\psi$. Thus, previous studies adopt an optimistic perspective. A key challenge of HUSM is that the $u_{\max}$ measure does not satisfy the downward closure property (DCP), a powerful property for efficiently reducing the search
space in FSM. To address this problem, many researchers proposed upper bounds (UB) on \( u_{\text{max}} \) that restore the DCP. The first such UB is the SWU (Sequence-Weighted Utility), which has been used in several algorithms [9, 12, 16]. It was demonstrated that all super sequences of a sequence having a low utility w.r.t. the SWU also have a low utility w.r.t. SWU and \( u_{\text{max}} \). This property is used to reduce the search space. However, since the SWU measure is not a tight UB on \( u_{\text{max}} \), the number of candidate sequences considered by SWU based algorithms can still be very large. To more efficiently reduce the search space, different UBs that are tighter than the SWU have been proposed, and were shown to improve the performance of \( \mathcal{HUSM} \) [3, 10, 13, 16]. More information about these UBs is provided in Section 3.

However, in real-world applications, using an optimistic perspective for measuring the utility can be risky. For example, consider that investors want to expand a business by opening new branches, and that the historical business information about the corporation is stored in a QSDB. Using the optimistic approach of previous work on \( \mathcal{HUSM} \) to analyze this database may result in taking unnecessary risks. The reason is that the utility of a sequence is calculated as the maximum utility among its occurrences. Thus, the utility of a pattern (sequence) may seem higher than it actually is in most input sequences. In fact, a sequential pattern may appear to have a high utility in some input quantitative sequences due to occurrences with large purchase quantities, while other occurrences of the pattern may have much smaller purchase quantities. Thus, measuring the utility of a pattern using the maximum utility of its occurrences is misleading, as it only provides information based on the largest purchase quantities in each input sequence. Hence, using such high utility patterns in investment can lead to taking important risks as it overestimates the quantities typically purchased by customers. To reduce risk, it is often desirable to use a more conservative approach by expecting that the minimum amount of utility is obtained from each input sequence. This paper thus proposes to calculate the utility of a pattern (sequence) using its minimum utility in each quantitative sequence (using a novel \( u_{\text{min}} \) measure) instead of its maximum utility (using \( u_{\text{max}} \)).

**Contributions.** The main contributions of this paper are the following. The problem of \( \mathcal{HUSM} \) with the novel \( u_{\text{min}} \) measure is formalized. Two new UBs named RBU and LRU on \( u_{\text{min}} \) are proposed, which are tighter than the traditional SWU UB. Moreover, two efficient strategies are designed for pruning the search space. Based on these strategies, a novel efficient algorithm named EHUSM is introduced for mining high utility sequences. Finally, an experimental evaluation is conducted on both real-life and synthetic QSDBs to assess the runtime and memory consumption of the proposed algorithm.

The rest of this paper is organized as follows. Section 2 introduces preliminary concepts. Section 3 presents the two novel UBs, pruning strategies and algorithm. Section 4 describes the experimental evaluation. Finally, Section 5 draws the conclusion.

2. Preliminary Concepts. This section introduces fundamental concepts and notations related to high utility sequence mining in quantitative sequence databases using the novel minimum utility measure \( u_{\text{min}} \), and the order relation on quantitative sequences and extensions of sequences.

**Definition 2.1.** (Quantitative sequence database). Let \( \mathcal{A} \equiv \{a_1, a_2, \ldots, a_M\} \) be a set of distinct items. Each item \( a_k \in \mathcal{A}, k = 1, \ldots, p \) is associated with a positive number called quality \( p(a_k) \) which represents its relative importance (e.g. unit profit). Item quality values are stored in a quality vector \( P(\mathcal{A}) \equiv (p(a_1), p(a_2), \ldots, p(a_M)) \). A quantitative item (or briefly q-item) is a pair \( (a, q) \), where \( a \in \mathcal{A} \) and \( q \in \mathbb{R}_+ \) is a positive number representing the purchase quantity of item \( a \). A subset \( E \) of \( \mathcal{A} \), \( E \subseteq \mathcal{A} \), is called an itemset or an event.
Without loss of generality, we assume that items in itemsets are sorted according to a total order such as the lexicographical order \( \prec \). A quantitative itemset (or \( q \)-itemset/\( q \)-event) according to \( E \) is denoted and defined as

\[
E' \overset{\text{def}}{=} \{(a_i, q_i) \mid a_i \in E, q_i \in R_+\}
\]

and \( E \) is called the projected itemset of \( E' \), which is denoted as \( E = \text{proj}(E') \). A list of \( q \)-itemsets, \( E_k', k = 1, \ldots, p \), where \( p \) is a positive integer, denoted as

\[
\alpha' = E_1' \rightarrow E_2' \rightarrow \ldots \rightarrow E_p'
\]
is called a \( q \)-sequence. Furthermore, we define \( \alpha'[k] \overset{\text{def}}{=} E'_k \). The projected sequence \( \alpha \) of a \( q \)-sequence \( \alpha' \) is defined as

\[
\alpha = \text{proj}(\alpha') \overset{\text{def}}{=} \text{proj}(E'_1) \rightarrow \text{proj}(E'_2) \rightarrow \ldots \rightarrow \text{proj}(E'_p)
\]

Moreover, we define \( \alpha[k] \overset{\text{def}}{=} \text{proj}(E'_k) \). A \( q \)-sequence is a null \( q \)-sequence, denoted as \( () \), if all its itemsets are empty. A quantitative sequence database (QSDB) \( D' \) consists of a finite number of input \( q \)-sequences,

\[
D' = \{\Psi_i, i = 1, \ldots, N\}
\]

and a quality vector \( P(A) \). The size of the \( q \)-sequence \( \alpha' \) is \( \text{size}(\alpha') = p \), and its length is \( \text{length}(\alpha') = \sum_{k=1}^p |E_k'| \), where \( |E_k'| \) is the number of items in itemset \( E \).

For the convenience of readers, Table 1 summarizes the notation used in the rest of this paper to denote \( (q-) \) items, \( (q-) \) itemsets, \( (q-) \) sequences and input \( q \)-sequences.

### Table 1. Notation.

<table>
<thead>
<tr>
<th>Type</th>
<th>Representation</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Item</td>
<td>Roman letter</td>
<td>( a, b, c )</td>
</tr>
<tr>
<td>q-item</td>
<td>(Roman letter, number)</td>
<td>( (a, 2), (b, 5), (c, 3) )</td>
</tr>
<tr>
<td>Event</td>
<td>Capitalized roman letter</td>
<td>( A, B, C )</td>
</tr>
<tr>
<td>q-Event</td>
<td>Capitalized roman letter followed by ‘(^{\prime})’</td>
<td>( A', B', C' )</td>
</tr>
<tr>
<td>Sequence</td>
<td>Greek letter</td>
<td>( \alpha, \beta, \gamma )</td>
</tr>
<tr>
<td>q-sequence</td>
<td>Greek letter followed by ‘(^{\prime})’</td>
<td>( \alpha', \beta', \gamma' )</td>
</tr>
<tr>
<td>Input sequence</td>
<td>Capiacitized Greek letter</td>
<td>( \psi, \psi_{\text{index}} )</td>
</tr>
<tr>
<td>Input q-sequence</td>
<td>Capiacitized Greek letter followed by ‘(^{\prime})’</td>
<td>( \psi', \psi'_{\text{index}} )</td>
</tr>
</tbody>
</table>

Let \( \alpha' = E'_1 \rightarrow E'_2 \rightarrow \ldots \rightarrow E'_p \), \( \beta' = F'_1 \rightarrow F'_2 \rightarrow \ldots \rightarrow F'_q \) be two arbitrary \( q \)-sequences, and \( \alpha = E_1 \rightarrow E_2 \rightarrow \ldots \rightarrow E_p \), \( \beta = F_1 \rightarrow F_2 \rightarrow \ldots \rightarrow F_q \) be respectively their projected sequences.

**Definition 2.2.** (Partial order relation \( \sqsubseteq \)) Consider any two \( q \)-itemsets

\[
E' = \{(a_{i_1}, q_{i_1}), \ldots, (a_{i_m}, q_{i_m})\}, \quad F' = \{(a_{j_1}, q_{j_1}), \ldots, (a_{j_n}, q_{j_n})\}, \quad m \leq n.
\]
The \( q \)-itemset \( E' \) is said to be contained in \( F' \) and denoted as \( E' \sqsubseteq F' \), iff (i.e. if and only if) there exist natural numbers \( 1 \leq k_1 < k_2 < \ldots < k_m \leq n \) such that \( a_{i_{k_l}} = a_{j_{k_l}} \) and \( q_{i_{k_l}} = q_{j_{k_l}}, \forall l = 1, \ldots, m \). Then, \( \alpha' \) is said to be contained in \( \beta' \) and denoted as \( \alpha' \sqsubseteq \beta' \) (or \( \beta' \) is called a super-\( q \)-sequence of \( \alpha' \)) iff \( p \leq q \) and there exist \( p \) positive integers,

\[
1 \leq j_1 < j_2 < \cdots < j_p \leq q : E'_k \sqsubseteq F'_{j_k}, \forall k = 1, \ldots, p
\]

and

\[
\alpha' \sqsubseteq \beta' \iff (\alpha' \sqsubseteq \beta' \land \alpha' \neq \beta').
\]
EHUSM: Mining High Pessimistic Utility Sequences

TABLE 2. Integrated QSDB $D'$

<table>
<thead>
<tr>
<th>SID</th>
<th>Input q-sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$\psi_1 = (a,2)(c,5)(e,6) \rightarrow (a,3)(b,6) \rightarrow (a,5)(d,50) \rightarrow (a,5)(b,9)(c,40) \rightarrow (a,4)(c,10)(d,10)(f,36)$</td>
</tr>
<tr>
<td>2</td>
<td>$\psi_2 = (b,12) \rightarrow (a,2)(c,20)(e,6) \rightarrow (a,3)(d,20) \rightarrow (a,1)(c,20)(d,10)(f,9) \rightarrow (a,4)(b,9)(c,15)$</td>
</tr>
<tr>
<td>3</td>
<td>$\psi_3 = (c,20) \rightarrow (a,4)(c,10)(e,4) \rightarrow (a,1)(f,18)$</td>
</tr>
<tr>
<td>4</td>
<td>$\psi_4 = (d,80) \rightarrow (a,7)(c,50)(e,6) \rightarrow (a,2)(g,2) \rightarrow (a,9)(f,72)$</td>
</tr>
</tbody>
</table>

Similarly, for simplicity, we also use $\sqsubseteq$ to define the containment relation over all sequences as follows: $\alpha \sqsubseteq \beta$ (or $\beta$ is a super-sequence of $\alpha$) $\iff \exists p$ positive integers,

$$1 \leq j_1 < j_2 < \ldots < j_p \leq q : E_k \subseteq F_{j_k}, \forall k = 1, \ldots, p$$

and

$$\alpha \sqsubseteq \beta \iff (\alpha \sqsubseteq \beta \land \alpha \neq \beta).$$

The q-sequence $\beta'$ contains the sequence $\alpha$ (or $\alpha$ is a sub-sequence of $\beta'$), denoted as $\alpha \sqsubseteq \beta'$ or $\beta' \sqsupseteq \alpha$, iff $\text{proj}(\beta') \sqsupseteq \alpha$.

Now, we let

$$\rho(\alpha) \overset{\text{def}}{=} \{ \psi' \in D' \mid \psi' \sqsupseteq \alpha \}$$

denote the set of all input q-sequences containing $\alpha$. The support of $\alpha$ is defined as the number of super-q-sequences of $\alpha$, that is

$$\text{supp}(\alpha) = |\rho(\alpha)|.$$

**Definition 2.3.** (Utilities of q-elements) The utilities of q-item $(a,q)$, q-event $E' = \{(a_{i_1},q_{i_1}),\ldots,(a_{i_m},q_{i_m})\}$, q-sequence $\alpha'$ and QSDB $D'$ are respectively defined and denoted as

$$u((a,q)) \overset{\text{def}}{=} p(a) \ast q, u(E') \overset{\text{def}}{=} \sum_{j=1}^{m} u((a_{i_j},q_{i_j})),$$

$$u(\alpha') \overset{\text{def}}{=} \sum_{i=1}^{p} u(E'_i), u(D') \overset{\text{def}}{=} \sum_{\psi' \in D'} u(\psi').$$

To avoid repeatedly calculating the utility $u$ of each q-item $(a,q)$ in all q-sequences $\psi'$ of $D'$, we calculate once all utility values, and replace $q$ by $u((a,q)) = p(a) \ast q$. This leads to an equivalent database representation of the QSDB $D'$ that is called the integrated QSDB of $D'$. For the sake of brevity, it is also denoted as $D'$. Hereafter, we only consider the integrated QSDB of $D'$. For example, Table 2 shows an integrated $D'$, which will be used as running example in this paper.

**Definition 2.4.** (Novel minimum utility measure of a sequence) Assume that $\alpha \sqsubseteq \beta'$. Let

$$U(\alpha,\beta') \overset{\text{def}}{=} \{ \alpha' \mid \alpha' \sqsubseteq \psi'_i \land \text{proj}(\alpha') = \alpha \}$$

be the set of all occurrences $\alpha'$ of $\alpha$ in $\beta'$. The minimum utility (or briefly utility) of $\alpha$ in $\beta'$ and in $D'$ are respectively defined as

$$u_{\text{min}}(\alpha,\beta') \overset{\text{def}}{=} \min\{ u(\alpha') \mid \alpha' \in U(\alpha,\beta') \},$$

and $u_{\text{min}}(\alpha,D')$ or

$$u_{\text{min}}(\alpha) \overset{\text{def}}{=} \sum_{\psi' \in \rho(\alpha)} u_{\text{min}}(\alpha,\psi').$$
The sequence \( \alpha \) is said to be a high utility (HU) sequence if its utility in \( D' \) is not less than a user-defined \( mu \) threshold, that is
\[
 u_{\text{min}}(\alpha) \geq mu.
\]
Otherwise, \( \alpha \) is said to be a low utility (LU) sequence. The problem of high utility sequence mining (HUSM) with \( u_{\text{min}} \) considered in this paper is to discover the set
\[
 \text{HUS} \text{ def } \{ \alpha \mid u_{\text{min}}(\alpha) \geq mu \}.
\]

For example, consider the sequence \( \alpha = d \to ac \to a, \rho(\alpha) = \{ \psi'_1, \psi'_2, \psi'_3 \} \). The sequence \( \alpha \) first appears in \( \psi'_3 \) as the sub-q-sequence \( \alpha' = (d, 80) \to (a, 7)(c, 50) \to (a, 2) \) of \( \psi'_4 \), \( \alpha = \text{proj}(\alpha') \) and \( u(\alpha') = 80 + 7 + 50 + 2 = 139 \). Hence, \( \alpha \subseteq \psi'_4 \), and \( \alpha' \in U(\alpha, \psi'_4) \).
Similarly, \( \alpha \) also appears in \( \psi'_4 \) as the sub-q-sequence \( \alpha'' = (d, 80) \to (a, 7)(c, 50) \to (a, 9) \) of \( \psi'_4 \) with the utility \( u(\alpha'') = 146 \). Hence, \( U(\alpha, \psi'_4) = \{ \alpha', \alpha'' \} \) and \( u_{\text{min}}(\alpha, \psi'_4) = \min\{u(\alpha'), u(\alpha'')\} = 139 \). In the same way, we also have \( u_{\text{min}}(\alpha, \psi'_4) = 99, u_{\text{min}}(\alpha, \psi'_2) = 45 \). Thus, \( u_{\text{min}}(\alpha) = u_{\text{min}}(\alpha, \psi'_4) + u_{\text{min}}(\alpha, \psi'_2) + u_{\text{min}}(\alpha, \psi'_3) = 99 + 45 + 139 = 283 \). For the running example and \( mu = 350 \), since \( \beta = d \to ac \to af \) with \( u_{\text{min}}(\beta) = 353 > mu \) is the unique HU sequence, thus \( \text{HUS} = \{ \beta \} \).

**Definition 2.5. (Extensions of a sequence).** The i-extension (or s-extension) of \( \alpha \) and \( \beta \) is defined and denoted as
\[
 \alpha \circ_i \beta = E_1 \to E_2 \to \ldots \to (E_p \cup F_1) \to F_2 \to \ldots \to F_q,
\]
where \( a < b, \forall a \in E_p, \forall b \in F_1 \) (or \( \alpha \circ_s \beta = E_1 \to E_2 \to \ldots \to E_p \to F_1 \to F_2 \to \ldots \to F_q \), respectively).

A forward extension (or briefly extension) of \( \alpha \) with \( \beta \), denoted as \( \gamma = \alpha \circ \beta \), can be either \( \alpha \circ_i \beta \) or \( \alpha \circ_s \beta \). To mine sequences, the search space of candidate sequences can be viewed as a prefix tree, which consists of the null sequence as its root, and where each node represents a candidate sequence, and each child node of a node is its extension. In the following, \( \text{branch}(\alpha) \) denotes the set consisting of \( \alpha \) and all its (forward) extensions.

Besides, any sequence \( \beta = \alpha \circ \varepsilon \circ y \) where \( \alpha \) is a non-null prefix, can be extended in a backward manner using a sequence \( \varepsilon \). The sequence \( \gamma = \alpha \circ \varepsilon \circ y \) such that \( \gamma \supseteq \beta \) is called a backward extension of \( \beta \) (by \( \varepsilon \) w.r.t. the last item \( y = \text{lastItem}(\beta) \)).

For instance, the sequence \( \alpha = be \), \( \text{branch}(\alpha) \) consists of sequence \( \alpha \) and its forward extensions \( be \to \varepsilon, bef \to \varepsilon, beg \to \varepsilon \), and \( befg \to \varepsilon, \forall \varepsilon \) (including \( \langle \rangle \) ). For \( \beta = be \), the backward extensions of \( \beta \) are \( bce, bde, bcd \), and backward extensions of \( b \to e \) are sequences \( bC \to \varepsilon \to e, \forall C \subseteq cdefg \) (e.g. \( bc \to \varepsilon \to e, bcd \to \varepsilon \to e \) and \( b \to \varepsilon \to De, \forall D \subseteq abed \) (e.g. \( b \to \varepsilon \to ace, b \to \varepsilon \to abde \)).

A main challenge of HUSM with \( u_{\text{min}} \) is that the \( u_{\text{min}} \) measure is neither monotonic nor anti-monotonic, i.e.
\[
 \exists (\beta \supseteq \alpha \land \gamma \subseteq \delta) \mid (u_{\text{min}}(\beta) > u_{\text{min}}(\alpha)) \land (u_{\text{min}}(\gamma) > u_{\text{min}}(\delta)).
\]

Thus, the DCP does not hold for \( u_{\text{min}} \). The DCP is a strong property used in FSM for efficiently pruning unpromising branches of the prefix tree (the search space). Indeed, for \( \alpha = ace \to a \to f, \beta = ace \to a \to af \) and \( \delta = ace \to a \to f \), then \( \delta \supseteq \alpha \subseteq \beta, u_{\text{min}}(\beta) = 243 > u_{\text{min}}(\alpha) = 229 > u_{\text{min}}(\delta) = 57 \). To address this problem, upper bounds on \( u_{\text{min}} \) satisfying anti-monotone-like properties that may be weaker than the DCP are proposed in the next section.

3. The Proposed Upper Bounds, Pruning Strategies And Algorithm.
3.1. Two Upper Bounds on \( u_{\min} \). Before presenting the two proposed upper bounds on \( u_{\min} \), we introduce the concept of ending and remaining \( q \)-sequence of a sequence in a \( q \)-sequence. Assume that \( \alpha' \subseteq \psi' \) and \( \alpha = \text{proj}(\alpha') \subseteq \psi' \), where \( \psi' = F_1' \rightarrow F_2' \rightarrow \ldots \rightarrow F_q' \in D' \), i.e. there exist \( p \) positive integers, \( 1 \leq i_1 < i_2 < \ldots < i_p \leq q \); \( E_k' \subseteq F_{i_k}' \) and \( E_k = \text{proj}(E_k') \subseteq \text{proj}(F_{i_k}') \), \( \forall k = 1, \ldots, p \). Then, the index \( i_p \) is said to be an ending of \( \alpha' \) (or \( \alpha \)) in \( \psi' \), denoted as \( \text{end}(\alpha', \psi') \) (or \( \text{end}(\alpha, \psi') \)). The last item of \( \alpha \) in \( F_{i_p}' \) is called an ending item according to \( i_p \) and is denoted as \( e_{i_p} \). Then, the \( \text{remaining} \) \( q \)-sequence of \( \alpha \) in \( \psi' \) w.r.t. the ending \( i_p \) is the rest of \( \psi' \) after the ending item \( e_{i_p} \) and is denoted as \( \text{rem}(\alpha, \psi', i_p) \). Let \( i_p^* = F \text{End}(\alpha, \psi') \) be the first ending of \( \alpha \) in \( \psi' \). If \( \alpha = \langle \rangle \), then as a convention, \( i_p^* = F \text{End}(\langle \rangle, \psi') \) and \( \text{rem}(\langle \rangle, \psi', i_p^*) = \psi' \). For each ending \( i_p = \text{end}(\alpha, \psi') \), we define

\[
 u(\alpha, \psi', i_p) \overset{\text{def}}{=} \min \{ u(\alpha') \mid \alpha' \in U(\alpha, \psi') \land \text{end}(\alpha', \psi') = i_p \} .
\]

Furthermore, we define

\[
 ub_{\text{rem}}(\alpha, \psi') \overset{\text{def}}{=} u(\alpha, \psi', i_p^*) + u(\text{rem}(\alpha, \psi', i_p^*))
\]

the remaining utility-based upper bound on \( u_{\min} \) of \( \alpha \) in \( \psi' \) for \( \alpha \neq \langle \rangle \), and \( ub_{\text{rem}}(\langle \rangle, \psi') \overset{\text{def}}{=} u(\psi') \) if \( \alpha = \langle \rangle \).

A measure \( \text{ub} \) is said to be an upper bound (UB) on \( u_{\min} \) iff \( u_{\min}(\alpha) \leq u(\alpha), \forall \alpha \).

For two UBs \( ub_1 \) and \( ub_2 \), \( ub_1 \) is said to be tighter than \( ub_2 \), denoted as \( ub_1 \ll ub_2 \), iff \( ub_1(\alpha) \leq ub_2(\alpha), \forall \alpha \), and \( ub_1 \) is said to be strictly tighter than \( ub_2 \), iff \( ub_1 \ll ub_2 \) and \( \exists \beta: ub_1(\beta) < ub_2(\beta) \).

For briefly, we set

\[
 ub_{\text{max}}(\alpha \circ \varepsilon) \overset{\text{def}}{=} \max \{ ub(\alpha \circ_i \varepsilon), ub(\alpha \circ_s \varepsilon) \},
\]

\[
 ub_{\text{max}}(\alpha \circ \varepsilon \circ \delta) \overset{\text{def}}{=} \max \{ ub(\alpha \circ \varepsilon \circ \delta), \forall \circ \in \{ \circ_i, \circ_s \} \}
\]

and define

\[
 IS_{ub}(\alpha) \overset{\text{def}}{=} \{ y \in A \mid ub_{\text{max}}(\alpha \circ y) \geq mu \}
\]

as the set of all candidate items for extensions of \( \alpha \) w.r.t. the \( \text{ub} \) UB on \( u_{\min} \), because if \( y \notin IS_{ub}(\alpha) \), then for \( \circ \in \{ \circ_i, \circ_s \} \), \( u_{\min}(\alpha \circ y) \leq ub_{\text{max}}(\alpha \circ y) < mu \).

Based on these concepts, we propose two novel UBs, named \( RBU \) and \( LRU \), which are tighter than the well-known SWU UB, i.e. \( RBU \ll LRU \ll SWU \). Although \( RBU \) is better than \( LRU \) in terms of values (i.e. providing a tighter UB), the width pruning effect of \( LRU \) is stronger than the depth pruning effect of \( RBU \).

**Definition 3.1. (Upper bounds on \( u_{\min} \))** Consider any item \( y \in A \) and non-null sequence \( \alpha \).

1. The traditional SWU UB (Sequence Weighted Utility) is defined [16] as

\[
 \text{SWU}(\alpha) \overset{\text{def}}{=} \sum_{\psi' \in \rho(\alpha)} u(\psi')
\]

2. Two novel UBs named RBU(Remaining-Based Utility) and LRU(Looser Remaining Utility) are defined as follows:

\[
 RBU(\alpha) \overset{\text{def}}{=} \sum_{\psi' \in \rho(\alpha)} ub_{\text{rem}}(\alpha, \psi'),
\]

\[
 \forall y \in A, \text{ if } \alpha \neq \langle \rangle,
\]

\[
 LRU(\alpha \circ y) \overset{\text{def}}{=} \sum_{\psi' \in \rho(\alpha \circ y)} ub_{\text{rem}}(\alpha, \psi'),
\]
else, obviously, $LRU(y) = SWU(y)$.

For example, consider the sequence $\beta = ae \rightarrow a \rightarrow c$, $\rho(\beta) = \{\psi_1', \psi_2'\}$. For $\psi_1'$, $u_{\min}(\beta, \psi_1') = 21$, since $\beta$ first appears in $\psi_1'$ as the sub-q-sequence $\beta' = (a,2)(e,6) \rightarrow (a,3) \rightarrow (c,40)$, the first ending of $\beta$ in $\psi_1'$ is $i_1^\ast \overset{\text{def}}{=} FEnd(\beta, \psi_1') = 4$, and the corresponding remaining q-sequence is $\text{rem}(\beta, \psi_1', i_1^\ast) = (a,4)(c,10)(d,10)(f,36)$, so $u(\text{rem}(\beta, \psi_1', i_1^\ast)) = 60$. Similarly, $\beta$ also appears in $\psi_1'$ as the sub-q-sequence $\beta'' = (a,2)(e,6) \rightarrow (a,5) \rightarrow (c,40)$, then $u(\beta, \psi_1', i_2^\ast) = \min\{u(\beta'), u(\beta'')\} = \min\{51,53\} = 51$ and $ub_{\text{rem}}(\beta, \psi_1') = u(\beta, \psi_1', i_2^\ast) + u(\text{rem}(\beta, \psi_1', i_2^\ast)) = 111$. In the same way, $u_{\min}(\beta, \psi_2') = 24$, $ub_{\text{rem}}(\beta, \psi_2') = 31 + 47 = 78$, so $u_{\min}(\beta) = u_{\min}(\beta, \psi_1') + u_{\min}(\beta, \psi_2') = 45$, $RUU(\beta) = ub_{\text{rem}}(\beta, \psi_1') + ub_{\text{rem}}(\beta, \psi_2') = 189$. Similarly, for $\alpha = ae \rightarrow a$, we also have $ub_{\text{rem}}(\alpha, \psi_1') = 186$ and $ub_{\text{rem}}(\alpha, \psi_2') = 99$. Hence, $\beta = \alpha \rightarrow c$, $LRU(\beta) = ub_{\text{rem}}(\alpha, \psi_1') + ub_{\text{rem}}(\alpha, \psi_2') = 285$, $SWU(\beta) = u(\psi_1') + u(\psi_2') = 191 + 131 = 322$. Hence, $u_{\min}(\beta) \leq RBU(\beta) \leq LRU(\beta) \leq SWU(\beta)$.

Before further discussing the properties of the UBs, we show that the $\rho$ operator and the $ub_{\text{rem}}$ UB on $u_{\min}$ are anti-monotonic in each input q-sequence.

**Lemma 3.1. (Properties of $\rho$ and $ub_{\text{rem}}$)**

a. (Anti-monotonicity of $\rho$) $\rho(\alpha) \supseteq \rho(\beta)$, $\forall \beta \sqsubseteq \alpha$.

b. (Anti-monotonicity w.r.t. forward extension of $ub_{\text{rem}}$ in each input q-sequence) $u_{\min}(\alpha, \psi') \leq ub_{\text{rem}}(\alpha, \psi')$, $\forall \psi' \in \rho(\alpha)$ and $ub_{\text{rem}}(\beta, \psi') \leq ub_{\text{rem}}(\alpha, \psi')$, $\forall \alpha = \epsilon \sqsubseteq \alpha$ and $\forall \psi' \in \rho(\beta)$.

**Proof:**

a. For any $\psi' \in \rho(\beta)$, then $\psi' \sqsubseteq \beta \sqsubseteq \alpha$ and hence $\psi' \in \rho(\alpha)$.

b. For any $\psi' \in \rho(\alpha)$, since $\{\alpha' \in U(\alpha, \psi') \wedge \text{end}(\alpha', \psi') = i_p^\ast\} \subseteq U(\alpha, \psi')$, $u_{\min}(\alpha, \psi') \leq u(\alpha, \psi', i_p^\ast) \leq u(\alpha, \psi', i_p^\ast) + u(\text{rem}(\alpha, \psi', i_p^\ast)) = ub_{\text{rem}}(\alpha, \psi')$, where $i_p^\ast \overset{\text{def}}{=} FEnd(\alpha, \psi')$.

For any $\psi' \in \rho(\alpha)$, define $\alpha'_{\text{min}}(\alpha, \psi')$ (or briefly $\alpha'_{\text{min}}$) as the q-sequence in $U(\alpha, \psi')$ that yields the minimum value $u(\alpha, \psi', i_p^\ast)$, that is $u(\alpha'_{\text{min}}) = u(\alpha, \psi', i_p^\ast)$, and $rem(\alpha', \psi')$ the rest of $\psi'$ after $\alpha'_{\text{min}}$. To prove that $ub_{\text{rem}}(\beta, \psi') \leq ub_{\text{rem}}(\alpha, \psi')$, for any forward extension of $\alpha = \beta \circ \delta \sqsubseteq \alpha$, without loss the generality, we can assume that the sequence $\delta$ only consists of an item $x$, i.e. $\delta = \beta \circ x$ and $\beta \overset{\text{def}}{=} \text{size}(\alpha)$. In the following, we denote the $k$th itemset of $\beta'$ as $\beta'[k]$ and the $k$th item of $\beta'[i]$ as $\beta'[i][k]$.

For any $\psi' \in \rho(\beta)$, consider the two following cases:

(i). $\beta = \alpha \circ x : \forall k : 1 \leq k \leq p$, $u(\beta'[k]) \leq u(\alpha'[k])$ and $u(\beta'[p + 1]) + u(\text{rem}(\beta'[p + 1])) \leq u(\text{rem}(\alpha'[k]))$. Thus, $ub_{\text{rem}}(\beta, \psi') = \sum_{1 \leq k \leq p} u(\beta'[k]) + u(\text{rem}(\beta'[p + 1])) \leq \sum_{1 \leq k \leq p} u(\alpha'[k]) + u(\text{rem}(\alpha'[k])) = ub_{\text{rem}}(\alpha, \psi')$.

(ii). $\beta = \alpha \circ x : \forall k : 1 \leq k \leq p - 1$, $u(\beta'[k]) \leq u(\alpha'[k])$ and $u(\beta'[p]) + u(\text{rem}(\beta'[p])) \leq u(\text{rem}(\alpha'[k]))$. Thus, $ub_{\text{rem}}(\beta, \psi') = \sum_{1 \leq k \leq p} u(\beta'[k]) + u(\text{rem}(\beta'[p])) \leq \sum_{1 \leq k \leq p} u(\alpha'[k]) + u(\text{rem}(\alpha'[k])) = ub_{\text{rem}}(\alpha, \psi')$.

The two following theorems show the relationships between the three aforementioned UBs in terms of their values and anti-monotone-like properties.

**Theorem 3.1. (Relationships between UBs on $u_{\min}$)**

$u_{\min} \ll RBU \ll LRU \ll SWU$.

Thus, $SWU$, $LRU$ and $RBU$ are gradually tighter UBs on $u_{\min}$.

**Proof:** For any forward extension, $\beta = \alpha \circ y \sqsubseteq \alpha$, $\alpha \neq \emptyset$ and $\psi' \in \rho(\beta)$, by Lemma 3.1, $ub_{\text{rem}}(\beta, \psi') \leq ub_{\text{rem}}(\alpha, \psi') \leq u(\psi')$. Thus, $RBU(\beta) \leq \sum_{\psi' \in \rho(\beta)} ub_{\text{rem}}(\alpha, \psi') = LRU(\beta) \leq SWU(\beta)$. 

Thus, we have $u_{\min} \ll RBU \ll LRU \ll SWU$. 

For any $\psi' \in \rho(\beta)$, consider the two following cases:

(i). $\beta = \alpha \circ x : \forall k : 1 \leq k \leq p$, $u(\beta'[k]) \leq u(\alpha'[k])$ and $u(\beta'[p + 1]) + u(\text{rem}(\beta'[p + 1])) \leq u(\text{rem}(\alpha'[k]))$. Thus, $ub_{\text{rem}}(\beta, \psi') = \sum_{1 \leq k \leq p} u(\beta'[k]) + u(\text{rem}(\beta'[p + 1])) \leq \sum_{1 \leq k \leq p} u(\alpha'[k]) + u(\text{rem}(\alpha'[k])) = ub_{\text{rem}}(\alpha, \psi')$.

(ii). $\beta = \alpha \circ x : \forall k : 1 \leq k \leq p - 1$, $u(\beta'[k]) \leq u(\alpha'[k])$ and $u(\beta'[p]) + u(\text{rem}(\beta'[p])) \leq u(\text{rem}(\alpha'[k]))$. Thus, $ub_{\text{rem}}(\beta, \psi') = \sum_{1 \leq k \leq p} u(\beta'[k]) + u(\text{rem}(\beta'[p])) \leq \sum_{1 \leq k \leq p} u(\alpha'[k]) + u(\text{rem}(\alpha'[k])) = ub_{\text{rem}}(\alpha, \psi')$. 

The two following theorems show the relationships between the three aforementioned UBs in terms of their values and anti-monotone-like properties.
SWU(β) and RBU(y) ≤ SWU(y) = LRU(y). Hence, RBU ≪ LRU ≪ SWU. Beside, for
i^*_p \overset{\text{def}}{=} FEnd(α, ψ'), we have \{α' ∈ U(α, ψ') ∧ end(α', ψ') = i^*_p \} ⊆ U(α, ψ'), \(u_{\min}(α, ψ') ≤ u(α, ψ', i^*_p) + u(rem(α, ψ', i^*_p)) = u_{\text{rem}}(α, ψ')\). Hence, \(u_{\min}(α) ≤ RBU(α)\) and \(u_{\min} ≪ RBU\).

Theorem 3.2. (Anti-monotone-like property of RBU, LRU and SWU).

a. SWU is anti-monotonic, denoted as \(\mathcal{AM}(SWU)\), i.e. \(SWU(α) ≥ SWU(β)\), for any super-sequence \(β\) of \(α\).

b. RBU is anti-monotonic w.r.t forward extensions, denoted as \(\mathcal{AMF}(RBU)\), i.e., \(RBU(α) ≥ RBU(β), ∀ forward\ extension\ β \in α\).

c. LRU is anti-monotonic w.r.t bi-directional (forward and backward) extensions, denoted as \(\mathcal{AMB}(LRU)\), i.e. \(\mathcal{AMF}(LRU)\) and \(LRU_{\max}(β) ≥ LRU_{\max}(γ), ∀ backward\ extension\ γ = α ∪ e ∪ y\ of β = α ∪ y\). Moreover, \(IS_{\text{LRU}}(α ∪ y) ≤ IS_{LRU}(α)\).

Proof:

a. \(∀β ≦ α, due to ρ(β) ⊆ ρ(α)\). Hence, \(SWU(β) ≤ SWU(α)\), i.e., \(AM(SWU)\).

b. Moreover, for any \(β = α ∪ e\) such that \(β \supseteq α\) and \(ψ' ∈ ρ(β) ⊆ ρ(α)\), by Lemma 3.1, then \(u_{\text{rem}}(β, ψ') ≤ u_{\text{rem}}(α, ψ')\). Hence, \(RBU(β) ≤ RBU(α), i.e., AMF(RBU)\).

c. To prove that \(LRU(β) ≤ LRU(α), ∀ β = α ∪ e ⊆ α\), without loss generality, we can assume that \(ε\) only consists of an item \(y\). Assume that \(α = δ ∩ x = δ ∩ x ∪ y\) with \(x, y ∈ \mathcal{A}\), then \(δ ∩ x\) is a forward extension of \(δ\). By Lemma 3.1, \(LRU(β) = \sum_{ψ' ∈ ρ(δ)} u_{\text{rem}}(δ ∩ x, ψ') ≤ \sum_{ψ' ∈ ρ(β)} u_{\text{rem}}(δ, ψ') < LRU(α)\).

Hence, \(AMF(LRU)\). To prove that \(LRU_{\max}(γ) ≤ LRU_{\max}(β), ∀ γ = α ∪ ε ∪ y\), \(β = α ∪ y\) such that \(γ ⊑ β\), first consider that \(γ = α ∪ ε) y\ and size(ε) = 1\. We have \(γ ⊑ β = α ∪ y\) and by Lemma 3.1, \(ρ(γ) ≤ ρ(α ∪ y)\). Hence, \(LRU(γ) = \sum_{ψ' ∈ ρ(γ)} u_{\text{rem}}(α ∪ ε, ψ') ≤ \sum_{ψ' ∈ ρ(β)} u_{\text{rem}}(α, ψ') ≤ \sum_{ψ' ∈ ρ(α ∪ y)} u_{\text{rem}}(α, ψ') = LRU(α ∪ γ)\).

For all other cases, for each \(γ ∈ \{α ∪ ε ∪ y, α ∪ ε ∪ ε ∪ y\}, \) we have \(α ∪ y \subseteq γ\). Thus, \(ρ(γ) ≤ ρ(α ∩ y)\) and \(LRU(γ) = \sum_{ψ' ∈ ρ(γ)} u_{\text{rem}}(α ∪ ε, ψ') ≤ \sum_{ψ' ∈ ρ(β)} u_{\text{rem}}(α, ψ') ≤ \sum_{ψ' ∈ ρ(α ∩ y)} u_{\text{rem}}(α, ψ') = LRU(α ∩ y)\).

Hence, \(∀ ε ∈ α ∪ ε ∪ y, LRU(γ) ≤ \sum_{ψ' ∈ ρ(α ∩ y)} u_{\text{rem}}(α, ψ') = LRU_{\max}(β)\).

For any \(z ∈ IS_{\text{LRU}}(α ∪ ε ∪ y), by the above proof, \(mu ≤ LRU_{\max}(α ∪ ε ∪ y) ≤ LRU_{\max}(α ∪ ε ∪ z), i.e., z ∈ IS_{\text{LRU}}(α)\).

Obviously, the anti-monotone-like properties \(AM, AMB\) and \(AMF\) are gradually weaker, i.e. \(AM(ub) → AMB(ub) → AMF(ub)\), for any UB \(ub\) on \(u_{\min}\). For example, \(AMB(LRU) → AMF(LRU), AM(SWU) → AMB(SWU) → AMF(SWU)\).

3.2. Two Upper Bounds on \(u_{\min}\). Based on Theorem 3.2, we design two strategies for pruning low utility candidate branches of the prefix search tree. These strategies only prune LU sequences, which ensures that all HU sequences can be found. Consider any item \(y ∈ \mathcal{A}\) and non-null sequence \(α\).

Depth pruning strategy based on RBU UB (DPS(RBU)): If \(RBU(α) < mu\), then we can prune the branch(α), because \(u_{\min}(β) ≤ RBU(β) ≤ RBU(α) < mu\, for any forward extensions \(β\) of \(α\).

Width pruning strategy based on LRU or SWU UBs (WPS(LRU) or WPS(SWU)): If \(LRU(α ∪ y) < mu\) or \(SWU(α ∪ y) < mu\), then we can prune not only all forward extensions of \(α ∪ y\) (or branch(α ∪ y)) but also all its backward extension branches by applying \(IS_{\text{LRU}}(α ∪ y) ⊆ IS_{\text{LRU}}(α)\).

Thus, if \(LRU(y) < mu\) (or equivalently \(SWU(y) < mu\)), then we can remove such irrelevant item \(y\) from QSDB \(D'\), because \(u_{\min}(γ) ≤ SWU(γ) ≤ SWU(y) = LRU(y) <
for any super-sequence $\gamma$ of $y$. Then, all UBs values according to the updated $D'$ should be also updated and may be decreased.

Similarly, if $LRU_{\text{max}}(\alpha \circ y) < mu$, such irrelevant item $y$ can be eliminated from the projected database $D'$ of $\alpha$. The reason is that for any sequence $\gamma = \alpha \circ \epsilon \circ y \circ \delta$ having the same prefix $\alpha$, which represents an arbitrary sequence in the PDB $D'_\alpha$ containing $y$, we always have $u_{\text{min}}(\gamma) \leq LRU(\gamma) \leq LRU(\alpha \circ \epsilon \circ y) \leq LRU_{\text{max}}(\alpha \circ \epsilon \circ y) \leq LRU_{\text{max}}(\alpha \circ y) < mu$. In other words, any sequence $\gamma$ in $D'_\alpha$ containing $y$ is also LU. In this context, the projected database $D'_\alpha$ of $\alpha$ consists of all remaining sequences $\text{rem}(\alpha, \psi', i'_p)$, where $\psi' \in \rho(\alpha)$ and $i'_p = \text{FEnd}(\alpha, \psi')$ is the first ending of $\alpha$ in $\psi'$. Then, all UBs values according to the reduced $D'_\alpha$ should be calculated and they can be reduced.

When extending a sequence $\beta = \alpha \circ y$ with an item $z$ from $\mathcal{A}$, we may generate candidate sequences that do not appear in $D'$. This results in unnecessary calculations. To address this problem, one can create a projected database (PDB) according to $\beta$. However, creating and scanning multiple PDBs is very costly. Based on the inclusion relationship $IS_{LRU}(\beta) \subseteq IS_{LRU}(\alpha)$ of Theorem 3.2.c, another way to overcome this problem is to only extend $\beta$ with an item $z$ (i.e. $z \in IS_{LRU}(\beta)$) if $z$ is in the candidate item set $IS_{LRU}(\alpha)$ of the previous step. Note that the $IS_{LRU}$ sets become gradually smaller during the mining process. Thus, this decreases the time for mining $\mathcal{HUS}$.

It is observed that $LRU(y) = SWU(y), \forall y \in \mathcal{A}$. The pruning effect of the RBU UB - the tightest UB among the three discussed UBs - is weaker than that of the LRU and SWU UBs (the pruning effect of the two latter UBs is stronger). In general, an UB $ub$ that is tighter than some others can be applied more often (when $ub(\alpha) < mu$) but its pruning effect is weaker. Beside, since both LRU and SWU UBs have the same width pruning effect, but LRU is tighter than SWU, LRU is always better than the SWU UB. Thus, to efficiently prune HU candidate sequences, it is sufficient to use the two proposed tighter LRU and RBU UBs.

For example, for $mu = 350$, since $SWU(b) = LRU(b) = 322, SWU(d) = LRU(d) = 228$ are less than $mu$, and the SWU and LRU values of remaining items in the candidate item set $IS = \{a, c, d, e, f\}$ are greater than $mu$ (e.g. $SWU(a) = LRU(a) = 609$). Then, by $\mathcal{WPS}(SWU)$ or $\mathcal{WPS}(LRU)$, not only the items $b$ and $g$ can be removed from $D'$ but the upper bound values of remaining items can be decreased. For example, the new updated $LRU(a)$ is 569 ($< 609$). Similarly, because $LRU(ad) = LRU(a \rightarrow d) = 286 < mu$, $IS_{LRU}(a) = acef$ and by $\mathcal{WPS}(LRU)$, we can completely prune $\text{branch}(ad)$ and $\text{branch}(a \rightarrow d)$ by $\mathcal{DPS}(LRU)$, but also eliminate many of their backward extension branches such as $\text{branch}(acd), \text{branch}(ac \rightarrow d), \text{branch}(ac \rightarrow d), \text{branch}(a \rightarrow c \rightarrow d)$, and $\text{branch}(ace \rightarrow d)$, because $IS_{LRU}(ac), IS_{LRU}(ae), IS_{LRU}(a \rightarrow c)$ are subsets of $IS_{LRU}(a)$ and $IS_{LRU}(ace) \subseteq IS_{LRU}(ac) \subseteq IS_{LRU}(a)$. Thus, $\mathcal{WPS}$ is strictly stronger than $\mathcal{DPS}$. To illustrate the depth pruning effect of RBU, consider values $RBU(a \rightarrow f) = 164$ and $RBU(f) = 154$ which are less than $mu$. Using $\mathcal{DPS}(RBU)$, the branches $\text{branch}(a \rightarrow f)$ and $\text{branch}(f)$ are pruned. Although $RBU$ is tighter than LRU, RBU does not have the width pruning effect. Indeed, assume conversely that $\mathcal{WPS}(RBU)$. If we remove $f$ from $D'$, then the sequence $\alpha = d \rightarrow ac \rightarrow af$ containing $f$ with $u_{\text{min}}(\alpha) = 353 > mu$ which is a HU sequence, will not be found in the final $\mathcal{HUS}$ result set.

**Discussion.** Instead of using the minimum utility $u_{\text{min}}$ in $\mathcal{HUSM}$, previous studies considered the maximum utility measure $u_{\text{max}}$ by considering an optimistic perspective. This latter measure is defined as $u_{\text{max}}(\alpha) \overset{\text{def}}{=} \sum_{\psi' \in \rho(\alpha)} u_{\text{max}}(\alpha, \psi')$, where $u_{\text{max}}(\alpha, \psi') \overset{\text{def}}{=} \max\{u(\alpha') | \alpha' \in U(\alpha, \psi')\}$ ([3, 9, 10, 13, 16]). It is neither anti-monotonic nor monotonic. Thus, upper bounds on $u_{\text{max}}$ that satisfy anti-monotonic-like properties have been proposed, e.g. $SWU$ [9], $\text{SPU}$ [16], $\text{CRO-M}$ [3], $\text{MEU}$ and $\text{LEU}$ [10]. For example, $\text{MEU}(\alpha) \overset{\text{def}}{=} [\ldots]$
∑_ψ∈ρ(α) MEU(α, ψ') and for β = α ∘ y, LEU(β) \text{ def } = \sum_ψ∈ρ(β) MEU(α, ψ') if α ≠ \emptyset, and LEU(y) \text{ def } = SWU(y) if α = \emptyset, where MEU(α, ψ') = \max\{u(α') + u(rem(α', ψ, i_p)) | ∀α' ∈ U(α, ψ'), i_p = end(α', ψ')\} [10]. Then, u_{min} ≤ u_{max} ≤ MEU ≤ LEU ≤ SWU, RBU and LRU UBs on u_{min} are strictly tighter than MEU and LEU UBs on u_{max}, respectively. Obviously, the number of high utility sequences using u_{max} is not less than that when using u_{min}.

For example, consider another integrated QSDB D'' = \{(b, 1) → (b, 2) → (d, 1) → (b, 80) → (c, 2) → (c, 4)(e, 1) → (d, 3) → (c, 3)(e, 4), (a, 5), (e, 2) → (d, 4) → (c, 3)(e, 4)\} and mu = 50. Then, the sequence α = b → d is a HU sequence using u_{max} but not u_{min}, because u_{max}(α) = 83 ≥ mu > u_{min}(α) = 2, and the number of HU sequences using u_{max} or u_{min} are respectively 4128 or 2752.

For β = a → b → d → ce, we have u_{min}(β) = 9, RBU(β) = 23, LRU(β) = 25, u_{max}(β) = 90, MEU(β) = 90, LEU(β) = 96 and SWU(β) = 115. It is observed that u_{min}(β) < RBU(β) < LRU(β) < u_{max}(β) = MEU(β) < LEU(β) < SWU(β). Hence, RBU and LRU are only UBs on u_{min} but not on u_{max}.

Moreover, note that the measure SPU ([16, 17]) is not really an UB on MU. SPU UB can be defined as SPU(α) = \sum_ψ∈ρ(α)[ub_p(α, ψ') + u(rem(α, ψ', i_p))], where i_p = FEnd(α, ψ') is the first ending (or pivot) of α in ψ' and ub_p(α, ψ') = \max\{u(α') | ∀α' ∈ U(α, ψ') and end(α', ψ') = i_p\}. Then, SPU(β) = 24 < u_{max}(β).

Thus, the well-known USpan [16] algorithm for mining HUS using u_{max} and its SPU UB is an incomplete algorithm, i.e. it may miss some HU sequences. Similar, for the CRoM UB (see more details in [3]) and α = b → d, then CRoM(α) = 182, SWU(α) = 115. Hence, CRoM(α) > SWU(α) and the performance of the HuspExt [3] algorithm that relies on CRoM may decrease in such cases.

3.3. The EHUSM Algorithm. Based on the above theoretical results, we design an efficient algorithm named EHUSM (Efficient High Utility Sequence Mining) for mining the set HUS of all high utility sequences when considering the u_{min} measure. The pseudo-code is shown in FIGURE 1. It takes as input a QSDB D' and the mu threshold. At the first level of the prefix-tree, the algorithm applies \(\mathcal{WPS}(SWU)\) or \(\mathcal{WPS}(LRU)\). The algorithm scans QSDB D' once to calculate the set IS = \{i ∈ A | SWU(i) ≥ mu\} of HU candidate items. Then, all irrelevant items (items in A \ IS) are removed from D' (lines 1-2). Next, the procedure SearchHUS is called for each item i ∈ IS (line 3).

![Algorithm 1. EHUSM(D', mu)](image)

The recursive SearchHUS procedure (given in FIGURE 2) takes as input a sequence α, a candidate item set IS, and the mu threshold. The procedure applies the depth pruning strategy \(\mathcal{DPS}(RBU)\) in line 1. If u_{min}(α) ≥ mu, the HU sequence α is output (line 2).
Next, in lines 4-9, based on the inclusion \( \text{NewIS} = IS_{LRU}(\alpha \diamond i) \subseteq IS = IS_{LRU}(\alpha) \), the width pruning strategy \( \text{WPS}(LRU) \) is applied for extensions of \( \alpha \). Afterward, \( \text{DPS}(RBU) \) is used again in line 10. Finally, the SearchHUS procedure is recursively called for each item using the two candidate item sets \( \text{newI} \) and \( \text{newS} \) to perform i-extensions and s-extensions, respectively (lines 11-12). Theorems 3.1 and 3.2 guarantee the correctness of EHUSM using two strategies \( \text{WPS} \) and \( \text{DPS} \). They are based on \( LRU \) and \( RBU \) UBs and help to prune unpromising branches in the prefix-tree early without missing HU sequences.

The execution of EHUSM is illustrated as follows. For \( mu = 350 \), after performing line 1 of Algorithm 1, two irrelevant items \( b \) and \( d \) are removed from \( D' \) (line 2) because \( LRU(b) = SWU(b) = 322 \), \( LRU(d) = 228 \). Afterward, updated values of \( RBU \) and \( LRU \) of all remaining items in the set \( IS = \{a,c,d,e,f\} \) from the reduced database \( D' \) (without \( b \) and \( d \) ) may be also reduced. Indeed, for instance, consider the original QSDB \( D' \), where \( LRU(e) = SWU(e) = 609 \). In the reduced database \( D' \) after discarding two items \( b \) and \( d \), the new updated \( LRU(e) = 569 \) (\( < mu \)) is less than \( mu \) and branch \( (e) \) is pruned earlier than if \( D' \) was not reduced (line 10).

Now, we briefly show the execution of the \( \text{SearchHUS}(\alpha, IS) \) procedure. For the sequence \( \alpha = f \), since \( RBU(f) = 154 < mu \), branch \( (f) \) is deeply pruned (line 1). For the sequence \( \alpha = e \), \( LRU(e) = 569 > RBU(e) = 369 > mu > u_{\min}(e) = 22 \). Hence lines 1-2 are skipped. Moreover, by lines 3-8, because \( LRU(e \rightarrow f) = 0 \), \( LRU(e \rightarrow a) = LRU(e \rightarrow f) = 369 \), \( LRU(e \rightarrow a) = 369 \), \( LRU(e \rightarrow f) = 369 \), \( > mu \), \( e \) of \( e \) are less than \( mu \), for \( y \in \{c,d,e\} \), then \( \text{newI} = \emptyset \) and \( \text{newS} = af \) (\( \subseteq IS \)), so \( \text{newI} = \emptyset \) and \( \text{newS} = af \). After deleting all irrelevant items \( c,d,e \) from \( D' \), then we obtain the reduced \( D'_e = \{(a,3) \rightarrow (a,5) \rightarrow (a,5) \rightarrow (a,4)(f,36),(a,3) \rightarrow (a,1)(f,9) \rightarrow (a,4),(a,1)(f,18),(a,2) \rightarrow (a,9)(f,72)\} \). Hence, the updated value \( LRU(e) = 194 \) (\( < 569 \)) is less than \( mu \) and branch \( (e) \) is pruned earlier than if \( D'_e \) was not reduced (line 10). \( \text{SearchHUS} \) is applied for other sequences in a similar way. Finally, EHUSM returns only one HU sequence \( \alpha = d \rightarrow ac \rightarrow af \), with \( u_{\min}(\alpha) = 353 \).
4. Experimental Evaluation. Experiments were performed on a computer having an Intel(R) Core (TM) i5-2320, 3.0 GHz CPU and 4 GB of memory, running Windows 8.1. EHUSM is the first algorithm for mining high utility sequences based on the minimum utility measure, $u_{\text{min}}$. It is implemented using the Java SE 1.8 programming language.

**Table 3. Parameters of the IBM Quest Synthetic Data Generator.**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>Number of sequences (in thousands) in the database</td>
</tr>
<tr>
<td>C</td>
<td>Average number of item-set per sequence</td>
</tr>
<tr>
<td>T</td>
<td>Average number of items per item-set</td>
</tr>
<tr>
<td>N</td>
<td>Number of different items (in thousands) in the database</td>
</tr>
<tr>
<td>S</td>
<td>Average number of item-sets in maximal sequences</td>
</tr>
<tr>
<td>I</td>
<td>Average number of items in maximal sequences</td>
</tr>
</tbody>
</table>

In the experiments, we consider two well-known benchmark real-life datasets named Gazelle and Snake [8], and three synthetic datasets generated using the IBM Quest data generator (obtained from [8]) using parameters described in Table 3. Characteristics of the considered datasets are shown in Table 4.

**Table 4. Characteristics of databases.**

<table>
<thead>
<tr>
<th>Database</th>
<th>#sequence</th>
<th>#item</th>
<th>avg. seg. Length</th>
<th>type of data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gazelle</td>
<td>59,601</td>
<td>497</td>
<td>2.51</td>
<td>web click stream</td>
</tr>
<tr>
<td>Snake</td>
<td>163</td>
<td>20</td>
<td>60.6</td>
<td>protein sequences</td>
</tr>
<tr>
<td>D4C7T5N5S6I4</td>
<td>4,000</td>
<td>5,000</td>
<td>28.68</td>
<td>Synthetic</td>
</tr>
<tr>
<td>D0.5C10T15N2S6I4</td>
<td>500</td>
<td>2,000</td>
<td>127.7</td>
<td>Synthetic</td>
</tr>
<tr>
<td>D5C10T5N5S6I4</td>
<td>5,000</td>
<td>5,000</td>
<td>42.9</td>
<td>Synthetic</td>
</tr>
</tbody>
</table>

4.1. Comparison of UBs. In a first experiment, we compare the pruning effect of the proposed RBU and LRU UBs with the well-known SWU UB on two QSDBs, the real-life Gazelle and the synthetic D4C7T5N5S6I4 datasets. In this experiment, the $mu$ threshold is varied and the runtime and number of candidate sequences are measured. Six versions of the EHUSM algorithms are compared, respectively using the SWU, LEU, MEU, RBU, LRU, and both the RBU and LBU UBs, to reduce the search space.

Results are shown in FIGURE 3. It is observed that using only the RBU or LRU UB is usually better than using the SWU, MEU and LEU because much less candidate sequences (#Candidates) are generated. Moreover, it is observed on D4C7T5N5S6I4 that using LRU can result in generating less candidates than when using RBU for high $mu$ values (e.g. higher than 0.44%). The reason is that the pruning condition ($LRU(\alpha) < mu$) of LRU can be frequently applied and the width pruning effect of LRU is stronger than the depth pruning effect of RBU.

For low $mu$ thresholds (e.g. smaller than 0.44%), RBU is better than LRU since the pruning condition ($RBU(\alpha) < mu$) of RBU is applied more often than LRU, although the pruning effect of RBU is weaker. The version of EHUSM using both RBU and LRU is always faster by 8,2 and 2 (or 24, 4 and 4) times on average compared to the version using only the SWU, MEU and LEU on the D4C7T5N5S6I4 (or Gazelle) dataset, respectively.

On overall, these results show that the two proposed upper bounds and corresponding pruning strategies are very effective for pruning the search space and that they should be used together. Thus, for the following experiments, we only consider the version of
Figure 3. Pruning effect of UBs.

4.2. Influence of the mu parameter. The second experiment assesses the influence of the mu parameter on time, memory consumption and number of discovered HU sequences (#HUS) for the datasets of Table 4.

Results depicted in FIGURE 4 show that as mu is decreased, the runtime of EHUSM increases and memory usage does not significantly change on Gazelle and D4C7T5N5S6I4 for low database density $\delta \triangleq T/N = 0.1\%$. In particular, for very low mu values on Gazelle (from 6% to 2%) and D4C7T5N5S6I4 (from 2.9% to 2%), although the number of HU sequences (#HUS) increases, execution time and memory usage only slightly increase. On the dense Snake and D0.5C10T15N2S6I4 databases (with a high density of $\delta = 0.75\%$),
when \( mu \) is set to very high values (e.g. \( mu = 15.5\% \) for Snake and \( mu = 1.9\% \) for D0.5C10T15N2S6I4) few patterns are found (e.g. \#HUS = 1). However, the number of HU sequences increases very quickly when \( mu \) is decreased (e.g. for \( mu = 0.0001\% \) on D0.5C10T15N2S6I4), while the runtime slightly fluctuates.

4.3. Influence of database parameters. To evaluate the influence of database characteristics on the efficiency of EHUSM, a third experiment was performed where parameters of synthetic datasets (see Table 3) were varied. In the following, a parameter letter followed by the wildcard (*) symbol indicates that the parameter is varied for that dataset. FIGURE 5 shows how the time, memory consumption, and number of HU candidates (\#Candidates) generated by EHUSM vary when dataset parameters are increased. It is observed that the algorithm has nearly linear scalability in terms of runtime.

Overall, the above experiments have shown that the proposed algorithm is efficient in terms of runtime and memory usage for both real-life and synthetic datasets. It is also
observed that using simultaneously the two novel RBU and LRU UBs in EHUSM is much more efficient than using the traditional SWU UB.

5. Conclusions. This paper proposed a novel utility measure $u_{\text{min}}$ to evaluate sequences in QSDBs. Moreover, two new upper bounds on $u_{\text{min}}$ named RBU and LRU are proposed, which are tighter than the traditional SWU UB. Two depth and width pruning strategies are developed based on the two UBs to prune low utility branches of the prefix search tree early. Based on these strategies, an efficient algorithm named EHUSM has been designed for mining the set of high utility sequences $\mathcal{HUS}$. To our best knowledge, this is the first algorithm for mining $\mathcal{HUS}$ based on $u_{\text{min}}$. Experiments conducted on both real-life and synthetic QSDBs have shown that using the two proposed UBs considerably decreases the number of candidate sequences, as well as the runtime and memory consumption of EHUSM.
The two novel upper bounds and corresponding pruning strategies proposed in this paper could be applied to mine concise representations of HUS such as the sets of all closed and generator high utility sequences. Mining generator and closed sequences is useful as they often have very small cardinalities and summarize the set of HUS. Extending EHUSM for these problems will be considered in future work.

Acknowledgment. This work is funded by Vietnam’s National Foundation for Science and Technology Development (NAFOSTED) under grant number 102.05-2017.300.

REFERENCES


Tin Truong is a professor at the Department of Mathematics and Computer Science, University of Dalat, Vietnam. He received his B.S. degree in Mathematics from Dalat University in 1983 and his Ph.D. in Stochastic Optimal Control in 1990 at the Vietnam National University, VNU-Hanoi, Vietnam. His current research interests are artificial intelligence and data mining.

Anh Tran is working at Department of Mathematics and Computer Science, University of Dalat, Dalat, Vietnam. He received his BSc in 1999, MSc in 2004 and PhD in 2016. His interesting researches include data mining and artificial intelligent.

Hai Duong received his M.S degree in 2009 and his Ph.D degree in 2020 at the University of Science – VNU-HCMC, Vietnam. He is currently a lecturer and a researcher at the Department of Mathematics and Computer Science, University of Dalat, Vietnam.
Bac Le received his B.S. degree in Mathematics in 1984 and the M.S. degree in Computer Sciences in 1990. He received his Ph.D. in Mathematics for Computers and Computing Systems in 2000 at the University of Science - VNU-HCMC, Vietnam. He is currently the Head of Department of Computer Science as well as the Vice Dean, Faculty of Information Technology. His current research interests include artificial intelligence, soft computing, data mining and data science.

Philippe Fournier-Viger is a full professor at the School of Natural Sciences and Humanities, Harbin Institute of Technology (Shenzhen), China, and adjunct professor at the Department of Computer Science, University of Moncton, Canada. He received the Ph.D. in Computer Science, at the University of Quebec in Montreal, Canada in 2010. He is the founder of the popular SPMF data mining library. His research interests include data mining, pattern mining, sequence mining, sequence prediction, cognitive modeling, and real-life applications in industrial design.