High Utility Itemset Mining over Uncertain Datasets Based on a Quantum Genetic Algorithm

Ju Wang[1], Fuxian Liu[1] and Chunjie Jin[1]
[1] Air Force Engineering University
Xi'an, 710051 - China
[e-mail: yonglingjuke@outlook.com]
*Corresponding author: Ju Wang

Received August 27, 2017; revised February 28, 2018; accepted March 19, 2018; published August 31, 2018

Abstract

The discovered high potential utility itemsets (HPUIs) have significant influence on a variety of areas, such as retail marketing, web click analysis, and biological gene analysis. Thus, in this paper, we propose an algorithm called HPUIM-QGA (Mining high potential utility itemsets based on a quantum genetic algorithm) to mine HPUIs over uncertain datasets based on a quantum genetic algorithm (QGA). The proposed algorithm not only can handle the problem of the non-downward closure property by developing an upper bound of the potential utility (UBPU) (which prunes the unpromising itemsets in the early stage) but can also handle the problem of combinatorial explosion by introducing a QGA, which finds optimal solutions quickly and needs to set only very few parameters. Furthermore, a pruning strategy has been designed to avoid the meaningless and redundant itemsets that are generated in the evolution process of the QGA. As proof of the HPUIM-QGA, a substantial number of experiments are performed on the runtime, memory usage, analysis of the discovered itemsets and the convergence on real-life and synthetic datasets. The results show that our proposed algorithm is reasonable and acceptable for mining meaningful HPUIs from uncertain datasets.

Keywords: High utility itemsets; uncertain datasets; Quantum genetic algorithm; High potential utility itemsets; HPUIM-QGA

http://doi.org/10.3837/tiis.2018.08.005 **ISSN : 1976-7277**

1. Introduction

Knowledge discovery in datasets (KDD) is an emerging issue since important, implicit, unknown, and potentially useful information can be found from enormous datasets [1]. Frequent itemset mining (FIM), which is used to mine those itemsets that have occurrence frequencies that are no smaller than the minimum support threshold, is one of the most important and common tasks of data mining [2]. Apriori [3], which is based on breadth-first search, and FP-growth [4], which is based on depth-first search, are well-known fundamental FIM algorithms. However, these traditional FIM algorithms consider only whether the items are present or not in the transactions and assign the same profit value for every item. In real-life applications, several other factors, such as the quantity and cost, must be included in the FIM to discover the more valuable itemsets that allow retailers and managers to adopt more profitable business strategies [5]. For example, a shopper buys milk every month or every few weeks, but he only buys a computer every few years. There is no doubt that the frequency of buying milk is larger than that of buying a computer. However, it is difficult to say which profit is larger for the retailers, which is the key point from their perspective.

To address this problem, the issue of high utility itemsets mining (HUIM) is designed to mine itemsets that have utilities that are higher than the minimum utility threshold. In real-life applications, the utility of the itemsets can be defined by the users, such as the profit, weight and cost. Because the goal of HUIM is to identify items or itemsets in transactions that bring considerable profit to retailers, the discovered itemsets can be used in a variety of areas, such as retail marketing, web click analysis, and biological gene analysis [6]. Chan et al. [7] first developed a mining framework to discover the top-k closed utility itemsets. Then, Yao et al. [5] designed an approach to discover HUIs by considering the purchase quantity and the profit of the items. Since the above algorithms suffer from the problem of combinatorial explosion, Liu et al. [8] proposed a two-phase model, which can prune the unpromising HUIs in the early stage by the downward closure of the transaction weighted utility. Furthermore, Lin et al. [9] designed a high utility pattern (HUP) tree for mining HUIs, and many improved algorithms have been proposed based on the designed tree structure. Additionally, HUITWU [10] is a state-of-the-art algorithm for HUIM based on a tree structure, and it can derive a complete condensed representation of HUIs. In contrast to the traditional HUIM, Kannimuthua and Premalatha [11] first designed the GA-based algorithm to mine HUIs with a ranked mutation. The results show that it can handle large numbers of distinct items and transactions. However, a very large computation is required to set an appropriate initial population and the parameters of crossover and mutation that are required in the evolution process of the GAs. Then, Lin et al. [12] proposed a binary particle swarm optimization (PSO) approach to mining HUIs and an OR/NOR structure to reduce the invalid combinations for discovering HUIs. Comparison experiments show that the PSO-based algorithm has fewer parameters and better performance than the GA-based algorithm. These two algorithms achieve better performance in addressing the problem of HUIM based on evolutionary algorithms. However, most HUIM algorithms are developed to handle precise datasets, which ignore the existence probability of items or itemsets that could be introduced when data is collected from noisy data sources such as wireless sensors and WiFi systems.

In fact, for uncertain datasets, the itemsets with high utility and high existence probabilities are useful to users, not the itemsets with only one of these properties. For example, shopper A buys milk that has a quantity of 100, and its profit is 0.5 with a probability of 0.91, and shopper A buys a computer that with a quantity of 1, and its profit is 400 with a probability of 0.05. According to this situation, retailers could introduce a large amount of milk into their shops and reduce the number of computers, while the traditional utility of milk is smaller than that of a computer. To the best of our knowledge, Lin et al. [13] proposed PHUI-UP based on a two-phase model and PHUI-List based on a list structure, while Lan et al. [14] proposed UHUI-apriori based on Apriori, and these are the only algorithms that are used to solve the problem of HUIM over uncertain datasets. The main challenges of these approaches are the non-downward closure property and combinatorial explosion, especially when the number of items or transactions in the datasets are relatively large.

In this paper, to handle the problem of the non-downward closure property, an upper bound of the potential utility (UBPU) is developed to prune the unpromising itemsets in the early stage. To handle the problem of combinatorial explosion, evolutionary computation, which can find the optimal solutions quickly in very large datasets, is introduced. While HUPEumu-GRAM [11] and HUIM-BPSO [12] are proven to be efficient in discovering HUIs from condensed datasets, both of these approaches still have the problem of falling into partial optimal solutions, especially in the late stage of the evolution process and when the parameters are set incorrectly. The main reason for this problem arises from the diversity of population being not sufficient and having too many parameters that must be set. To handle these shortcomings, many new evolutionary algorithms have been proposed, such as the QGA [15], shuffled frog-leaping algorithm (SFLA) [16], improved discrete cuckoo search (IDCS) [17], and multi-population artificial bee colony (MPABC) [18]. Whereas the QGA is a relatively outstanding approach, its advancements are mainly reflected in the following aspects. First, the quantum chromosome can be used to characterize multiple states simultaneously, which implies strong parallelism and a better ability to maintain the diversity of the population. Second, the search process driven by the quantum rotation gate is the procedure that approaches the optimal solution according to the optimal individual's information. This method makes the population evolve to the best area with a great probability and accelerates the convergence speed. Third, to avoid local extrema, mutation is realized by the quantum non-gate. Fourth, when the size of the population and the maximum number of iterations are given, no other parameters are needed. Lastly, compared to traditional evolutionary computation, the QGA has a smaller population without affecting its performance. Therefore, aiming at the issue of HUIM over uncertain datasets, a QGA is introduced in this paper. The key contributions of this paper are described below.

1. Discovering HPUIs based on evolutionary computation over uncertain datasets is a relatively new issue. In this paper, a QGA-based algorithm, called HPUIM-QGA, is proposed to address this issue. It has fewer parameters that must be set, a better ability to maintain the diversity of the population, and a higher convergence speed.

2. A novel type of itemset called a high potential utility itemset (HPUI) is designed, which represents the itemsets that have high utility and a high existence probability.

3. Because potential utility does not hold the downward closure property in the mining process, an upper bound of the potential utility (UBPU) is developed to prune the un-useful itemsets in the early stage. This approach is an overestimation for the potential utility and can hold the downward closure property.

4. To avoid the meaningless and redundant itemsets that are generated in the evolution process of a QGA, a pruning strategy is proposed in this paper.

5. Extensive experiments are conducted on real-life and synthetic datasets to evaluate the performance of the proposed algorithm. The results show that the proposed algorithm can efficiently identify HPUIs over uncertain datasets.

The remainder of this paper is organized as follows. In section 2, we state the problem of HUIM over uncertain datasets and present our new definitions. In section 3, we develop our proposed HPUIM-QGA algorithm to address the stated problem. In section 4, our experimental results are presented and analyzed. Finally, in section 5, conclusions are drawn.

2. Preliminaries and problem statement

2.1 Preliminaries

$I = \{i_1, i_2, \cdots, i_L\}$ is a finite set of n distinct items in an uncertain dataset $D = \{T_1, T_2, \cdots, T_m\}$. A transaction $T_q = \{i_1, i_2, \cdots, i_l\}(1 \le l \le L)$ in D contains a number of items, and each item i_p in T_q is associated with an internal utility $iu(i_p, T_q)$ and existence probability $p(i_p, T_q)$, which indicates a quantity value of i_p in T_q and the likelihood of i_p being present in T_q, respectively [19]. In addition, the external utility of item i_p in I is represented by $eu(i_p)$, which indicates the profit value of i_p.

An example of uncertain datasets from a market is shown in **Table 1(a).** The external utility of the example is shown in **Table 1(b)**, which is also called a profit table.

Table 1(a). An example of uncertain datasets.

TID	Transaction
T_1	$(A, 1, 0.81)(C, 1, 0.79)(D, 2, 0.93)$
T_2	$(A, 1, 0.88)(C, 6, 0.92)$
T_3	$(A, 1, 0.83)(C, 3, 0.91)(D, 3, 0.86)(E, 1, 0.76)$
T_4	$(B, 4, 0.62)(C, 3, 0.89)(D, 3, 0.86)$
T_5	$(C, 2, 0.87)(F, 3, 0.63)$
T_6	$(A, 2, 0.93)(F, 4, 0.76)$

Table 1(b). Profit table of the example.

Item	A	B	C	D	E	F
Profit	3	6	5	8	4	3

Here, $(A, 1, 0.81)$ in the first row of **Table 1(a)** can be illustrated as a shopper T_1 with a probability of 0.81 buys item A in a quantity of 1, for which the profit is 3. It is clear that the itemsets with high probability and high utility are useful to researchers. Therefore, the aim of this paper is established to discover such itemsets. Related definitions are given below.

Definition 1 (Item utility). The utility of an item i_p in a transaction T_q is defined as the product of the external and internal utility of i_p, which is denoted as

$$u(i_p, T_q) = iu(i_p, T_q) \times eu(i_p).$$

For the example in **Table 1**, the item utility of $\{D\}$ in T_1 is calculated as $u(D, T_1) = iu(D, T_1) \times eu(D) = 2 \times 8 = 16$.

Definition 2 (Itemset utility in a transaction). The utility of an itemset X in a transaction T_q is defined as

$$u(X, T_q) = \sum_{i_p \in X} u(i_p, T_q).$$

For the example in **Table 1**, the utility of $\{AC\}$ in T_1 is calculated as $u(AC, T_1) = u(A, T_1) + u(C, T_1) = 3 + 5 = 8$.

Definition 3 (Transaction utility). The utility of a transaction T_q is defined as

$$tu(T_q) = \sum_{i_p \in T_q} u(i_p, T_q).$$

For the example in **Table 1**, the utility of T_1 is calculated as $tu(T_1) = u(A, T_1) + u(C, T_1) + u(D, T_1) = 3 + 5 + 16 = 24$.

Definition 4 (Itemset probability in a transaction). The probability of an itemset X in a transaction T_q is defined as

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

For the example in **Table 1**, the probability of $\{AC\}$ in T_1 is calculated as $p(AC, T_1) = p(A, T_1) \times p(C, T_1) = 0.81 \times 0.79 = 0.6399$.

Definition 5 (Potential utility of an itemset). Based on the expected support model, the potential utility of an itemset X is defined as

$$pu(X) = \sum_{T_q \in D \wedge X \subseteq T_q} u(X, T_q) \times p(X, T_q).$$

For the example in **Table 1**, the potential utility of $\{AC\}$ in D is calculated as $pu(AC) = u(AC, T_1) \times p(AC, T_1) + u(AC, T_2) \times p(AC, T_2) + u(AC, T_3) \times p(AC, T_3) = 8 \times 0.6399 + 33 \times 0.8096 + 18 \times 0.7553 = 45.4314$.

Definition 6 (High potential utility itemset (HPUI)). Given the minimum potential utility threshold δ ($\delta \in [0,1]$), an itemset X is defined as an HPUI when it satisfies

$$pu(X) \geq minpu = \delta \times \sum_{T_q \in D} tu(T_q).$$

For the example in **Table 1**, when $\delta = 0.2$, $minpu = 0.2 \times 203 = 40.6$, as $pu(AC) = 45.4314 > 40.6$, itemset $\{AC\}$ is a HPUI.

For the uncertain dataset in **Table 1**, given the minimum utility threshold $\delta_1 = 0.2$, according to the method in Ref. [8], **Table 2** shows the discovered HUIs. When the minimum utility threshold δ_1 is equal to the minimum potential utility threshold δ_2, **Table 3** shows the discovered HPUIs.

Table 2. The discovered HUIs when $\delta_1 = 0.2$.

Itemset	Utility	Itemset	Utility
$\{C\}$	75	$\{CD\}$	99
$\{D\}$	64	$\{ACD\}$	66
$\{AC\}$	59	$\{BCD\}$	63
$\{AD\}$	46	$\{CDE\}$	43
$\{BD\}$	48	$\{ACDE\}$	46

Table 3. The discovered HPUIs when $\delta_2 = 0.2$.

Itemset	Potential utility	Itemset	Potential utility
$\{C\}$	67.25	$\{CD\}$	75.8007
$\{D\}$	56.16	$\{ACD\}$	41.5640
$\{AC\}$	45.4314		

From **Tables 2** and **3**, it can be seen that only a few HUIs are HPUIs, and these represent the itemsets that have high utility and high existence probability. For example, itemset $\{AD\}$ is an HUI, but it is not an HPUI since $pu(AD) = 33.5853 < 0.2 \times 203$. This phenomenon shows that fewer and more useful itemsets can be found when considering both the utility and existence probability.

2.2 Problem statement

For an uncertain dataset, given a predefined profit table and a user-specified minimum potential utility threshold δ, the problem of HUIM over uncertain datasets is to discover the HPUIs that have a potential utility that is no smaller than $minpu$.

3. HUIM over uncertain datasets based on the QGA

The designed QGA-based algorithm for mining HPUIs over uncertain datasets is composed of pre-processing, chromosome encoding, an updating process, a pruning strategy, and a fitness evaluation. In the pre-processing phase, 1-SHPUIs (the supersets for high potential utility itemsets) are discovered according to the two-phase model, which can greatly prune the unpromising itemsets based on UBPU's downward closure property. In the chromosome encoding phase, an itemset in the transactions corresponds to a chromosome, which is encoded by quantum bits in alphabetic ascending order, and its length is decided by the number of 1-SHPUIs. In the updating phase, a quantum rotation gate is introduced to update the quantum bits of the chromosomes. Then, a pruning strategy is designed to avoid the meaningless and redundant itemsets that are generated in the updating phase. In the fitness evaluation phase, the potential utility is regarded as the fitness function, which is used to evaluate the chromosomes. If the fitness value of the chromosome is higher than the minimum potential utility value, it is considered to be an HPUI and is placed in the set of HPUIs. These phases are repeated until the termination criteria are achieved. The algorithm flow of the designed algorithm is described in **Fig. 1**, and the details are described below.

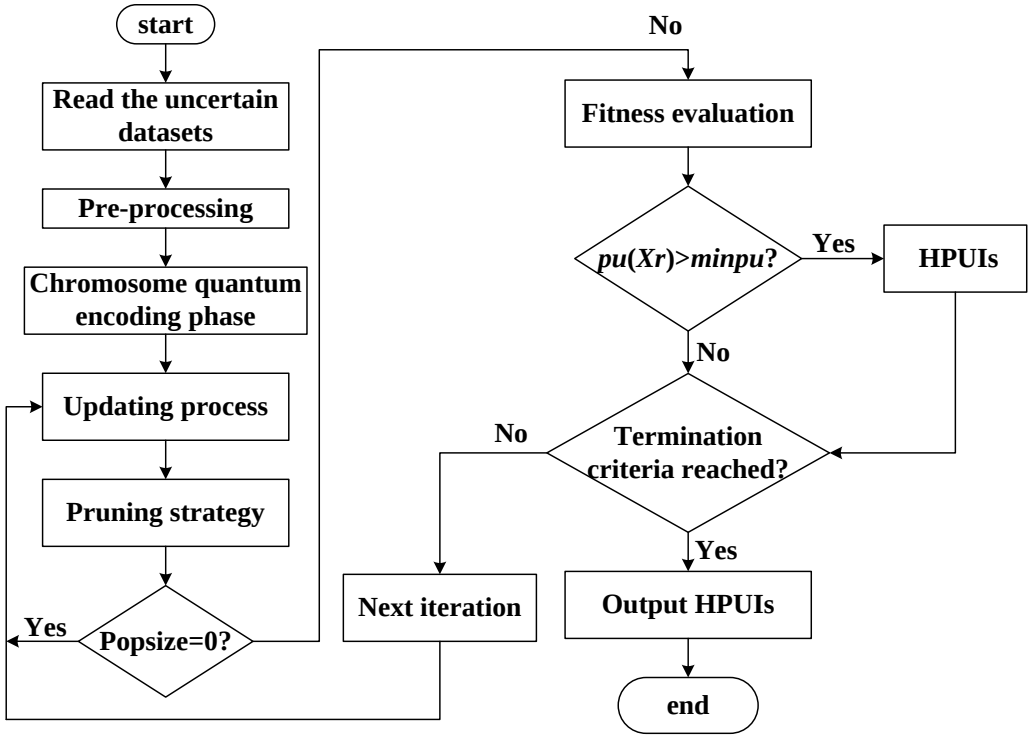


Fig. 1. Algorithm flow of the proposed HPUIM-QGA

3.1 Pre-processing

From **Tables 2** and **3**, it can be seen that itemset $\{AC\}$ is an HUI and HPUI, but item $\{A\}$ is not. This phenomenon shows that both the utility and potential utility do not have the downward closure property [3], which means that the potential utility constraint is neither monotone nor anti-monotone. Hence, unlike frequent itemset mining, the potential utility of an itemset cannot be used to prune the search space. In this paper, to address this problem, an upper bound of the potential utility (UBPU) is designed, which is an overestimation of the potential utility of the itemsets, and it can hold the downward closure property. UBPU will prune the un-useful itemsets in the early stage and perform the mining process of the HPUIs efficiently.

Definition 7 (The upper bound of the transaction probability). The upper bound of the transaction probability for transaction T_q is defined as

$$tp(T_q) = \max_{i_p \in T_q} p(i_p, T_q).$$

For the example in **Table 1**, the upper bound of the transaction probability for T_1 is calculated as $tp(T_1) = \max\{p(A,T_1), p(C,T_1), p(D,T_1)\} = \max\{0.81, 0.79, 0.93\} = 0.93$.

Theorem 1 ($p(X,T_q) \le tp(T_q)$). The probability of an itemset X in a transaction T_q is no larger than the existence probability of the transaction T_q, which is denoted as $p(X,T_q) \le tp(T_q)$.

Proof. Given

$$p(X,T_q)=\prod_{i_p\in T_q} p(i_p,T_q)\le p(i_p,T_q)\le \max_{i_p\in T_q} p(i_p,T_q)=tp(T_q).$$

Therefore, $p(X,T_q)\le tp(T_q)$.

Theorem 2 ($u(X,T_q)\le tu(T_q)$). The utility of an itemset X in a transaction T_q is no larger than the transaction utility of the corresponding transaction T_q, which is denoted as $u(X,T_q)\le tu(T_q)$.

Proof. Given

$$u(X,T_q)=\sum_{i_p\in X} u(i_p,T_q)\le \sum_{i_p\in T_q} u(i_p,T_q)=tu(T_q).$$

Therefore, $u(X,T_q)\le tu(T_q)$.

Definition 8 (The UBPU for an itemset). The UBPU for an itemset X in D is defined as

$$ubpu(X)=\sum_{X\subseteq T_q\wedge T_q\in D} tu(T_q)\times tp(T_q).$$

For the example in **Table 1**, the UBPU for $\{AC\}$ in D is calculated as $ubpu(AC)=24\times 0.93+33\times 0.92+46\times 0.91=94.54$.

Definition 9 (The supersets for high potential utility itemsets (SHPUIs)). Given the minimum potential utility threshold δ ($\delta\in[0,1]$), an itemset X is defined as an SHPUI when it satisfies

$$ubpu(X)\ge minpu=\delta\times\sum_{T_q\in D} tu(T_q).$$

For the example in **Table 1**, when $\delta=0.2$, $minpu=0.2\times 203=40.6$, and when $ubpu(AC)=94.54>40.6$, $\{AC\}$ is an SHPUI. While $ubpu(AF)=16.74<40.6$, $\{AF\}$ is not an SHPUI.

Theorem 3 (The downward property of UBPU). For an uncertain dataset D, X^{k-1} is a subset of X^k, where the length of X^{k-1} and X^k is $k-1$ and k, respectively. If X is an SHPUI, then any subset of it is also an SHPUI, which is denoted as $ubpu(X^k)\le ubpu(X^{k-1})$.

Proof. Since $X^{k-1}\subseteq X^k$, the set of $TIDs(X^k\subseteq T_q)\subseteq TIDs(X^{k-1}\subseteq T_q)$. Moreover, $tu(T_q)$ and $tp(T_q)$ are positive numbers according to their definitions. Thus,

$$\begin{aligned} ubpu(X^k)&=\sum_{X^k\subseteq T_q\wedge T_q\in D} tu(T_q)\times tp(T_q)\\ &\le \sum_{X^{k-1}\subseteq T_q\wedge T_q\in D} tu(T_q)\times tp(T_q)\\ &= ubpu(X^{k-1}).\end{aligned}$$

Therefore, if X^k is an SHPUI, then any subset of it is also an SHPUI.

Theorem 4 ($HPUIs \subseteq SHPUIs$). For an uncertain dataset D, if an itemset X is not an SHPUI, then no superset of X is an HPUI.

Proof. From Theorems 1 and 2, we can obtain that $p(X, T_q) \leq tp(T_q)$ and $u(X, T_q) \leq tu(T_q)$, respectively. Moreover, $tu(T_q)$ and $tp(T_q)$ are positive numbers, and thus,

$$\begin{aligned} pu(X) &= \sum_{X \subseteq T_q \wedge T_q \in D} u(X, T_q) \times p(X, T_q) \\ &\leq \sum_{X \subseteq T_q \wedge T_q \in D} tu(T_q) \times tp(T_q) \\ &= ubpu(X). \end{aligned}$$

Therefore, for an uncertain dataset D, if an itemset X is not an SHPUI, then no superset of X is an HPUI.

Consider the example in **Table 1**, when the minimum potential utility threshold δ_2 is set to 0.2. **Table 4** shows the discovered SHPUIs.

Table 4. The discovered SHPUIs when $\delta_2 = 0.2$.

Itemset	$ubpu$	Itemset	$ubpu$
$\{A\}$	111.28	$\{BD\}$	56.07
$\{B\}$	56.07	$\{CD\}$	120.25
$\{C\}$	167.35	$\{CE\}$	41.86
$\{D\}$	120.25	$\{DE\}$	41.86
$\{E\}$	41.86	$\{ACD\}$	64.18
$\{AC\}$	94.54	$\{ACE\}$	41.86
$\{AD\}$	64.18	$\{BCD\}$	56.07
$\{AE\}$	41.86	$\{ACDE\}$	41.86
$\{BC\}$	56.07		

From **Tables 3** and **4**, it can be seen that the HPUIs are included in the discovered SHPUIs under the same conditions. More importantly, it can be obtained that the proposed UBPU has the downward property. Therefore, to decrease the search space, 1-SHPUIs are discovered based on UBPU in the pre-processing phase of our proposed HPUIM-QGA algorithm. Its calculation process is described below.

First, calculate the UBPU of all of the items in the uncertain datasets according to Definition 8.

Second, given a minimum potential utility threshold δ according to the users' preference, i_p is considered to be a 1-SHPUI when $ubpu(i_p) \geq \delta \times \sum_{T_q \in D} tu(T_q)$.

For the example in **Table 1**, the UBPU of each item in it can be seen in **Table 5**.

Table 5. UBPU of the example.

Item	A	B	C	D	E	F
UBPU	111.28	56.07	167.35	120.25	41.86	25.65

Set the minimum potential utility threshold $\delta_2 = 0.2$, and we can obtain that $minpu = \sum_{T_q \in D} tu(T_q) \times \delta_2 = 203 \times 0.2 = 40.6$. Therefore, items A, B, C, D, E are 1-SHPUIs, and item F is pruned.

3.2 Chromosome encoding

Bit is a basic concept of classical information theory. In a quantum system, the information storage unit, similar to a physical medium, is in quantum bits. Compared with a classical bit, the state of a quantum bit $|\Psi\rangle$ can be any linear superposition state between '0' and '1', which is denoted as

$$|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle. \tag{1}$$

where $|\cdot\rangle$ is Dirac notation, and it is used to describe the state. $0\rangle$ and $1\rangle$ are the spin-down and spin-up state, respectively. α and β are two complex constants. $|\alpha|^2$ and $|\beta|^2$ are the probability of a quantum bit occurring in the '0' and '1' state, respectively, which satisfies $|\alpha|^2 + |\beta|^2 = 1$.

Based on the concept of quantum bits, a gene can be expressed and stored by one or more quantum bit(s). Furthermore, a chromosome consists of genes that can be expressed by several quantum bits. In this paper, an itemset in the transactions corresponds to a chromosome C_i, which is a set of quantum bits, and it is represented by

$$C_i = \begin{bmatrix} \alpha_{i1} & \cdots & \alpha_{in} \\ \beta_{i1} & \cdots & \beta_{in} \end{bmatrix}, 1 \le i \le sizepop. \tag{2}$$

where n is the number of 1-SHPUIs found in the pre-processing phase, $sizepop$ is the size of the population, which is set by the user, and $|\alpha_{ij}|^2 + |\beta_{ij}|^2 = 1, 1 \le j \le n$. In the very beginning, all of the $(\alpha_{ij}, \beta_{ij})$ are initialized as $(1/\sqrt{2}, 1/\sqrt{2})$.

Compared to a traditional encoding method, the state of every quantum bit is uncertainty, and it brings information in terms of different superposition states. Therefore, the quantum encoding method can make the population have more diversity. After a number of iterations, the value of $|\alpha|^2$ or $|\beta|^2$ in some quantum bit tends to 0 or 1, and the diversity that results from this uncertainty will gradually disappear, and then, the algorithm is convergent finally. This process shows that quantum encoding has the ability to explore and develop.

3.3 Updating process

Based on the theory of quantum mechanics and quantum computing, a QGA encodes a chromosome by using quantum bits with superposition states. Through quantum gates that act on quantum superposition states in quantum computing, the probability amplitudes of all of the states can be changed correspondingly, which realizes the updating process of the chromosomes. In the HPUIM-QGA, this process is completed by using the quantum rotation gate $U(\theta)$ described below, which is usually employed in the coding problems of 0 and 1.

$$U(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \tag{3}$$

where θ is the rotation angle, and the corresponding update operation of the quantum bits is described as

$$\begin{bmatrix}\alpha'_{ij}\\ \beta'_{ij}\end{bmatrix}=\begin{bmatrix}\cos(\theta_{ij}) & -\sin(\theta_{ij})\\ \sin(\theta_{ij}) & \cos(\theta_{ij})\end{bmatrix}\begin{bmatrix}\alpha_{ij}\\ \beta_{ij}\end{bmatrix}. \tag{4}$$

where $[\alpha_{ij},\beta_{ij}]^T$ is the j-th quantum bit of the i-th chromosome in the current population, θ_{ij} is the corresponding rotation angle, and its value $\Delta\theta_{ij}$ and sign $s(\alpha_{ij},\beta_{ij})$ are adjusted by the strategy shown in **Table 6**, which can make the population evolve to the optimal solution quickly by considering the information of the best individual.

Table 6. Selection strategy of rotation angle.

X_{ij}	$best_j$	$pu(X_i) > pu(best)$	$\Delta\theta_{ij}$	$s(\alpha_{ij},\beta_{ij})$			
				$\alpha_{ij}\beta_{ij}>0$	$\alpha_{ij}\beta_{ij}<0$	$\alpha_{ij}=0$	$\beta_{ij}=0$
0	0	F	0	0	0	0	0
0	0	T	0	0	0	0	0
0	1	F	0.01π	+1	-1	0	±1
0	1	T	0.01π	-1	+1	±1	0
1	0	F	0.01π	-1	+1	±1	0
1	0	T	0.01π	+1	-1	0	±1
1	1	F	0	0	0	0	0
1	1	T	0	0	0	0	0

In the **Table 6**, X_{ij} represents the measurement value of the j-th quantum bit in the i-th chromosome, and $best_j$ represents the measurement value of the j-th quantum bit in the current best chromosome. $\Delta\theta_{ij}$ is used to control the convergence speed of the QGA. When $\Delta\theta_{ij}$ is too small, the convergence speed becomes slow. When $\Delta\theta_{ij}$ is too large, the results could diverge or converge to the a locally optimal solution. Therefore, to search for the globally optimal solution, we set the value of $\Delta\theta_{ij}$ to 0.01π, which is a relatively small value. Here, $s(\alpha_{ij},\beta_{ij})$ is the direction of the rotation angle, which is important to the convergence of the algorithm.

The theory that the proposed strategy can make the population converge to the optimal solutions is stated as follows. When $X_{ij}=0$, $best_j=1$, and $pu(X_i)>pu(best)$, to make the current solution converge to a chromosome with higher fitness, the value of $|\alpha_{ij}|^2$ should be made larger. Therefore, if $\alpha_{ij}\beta_{ij}>0$, then θ rotates 0.01π clockwise. If $\alpha_{ij}\beta_{ij}<0$, then θ rotates 0.01π anticlockwise, as in the schematic diagram of the rotation gate in **Fig. 2**. Other situations are similar. It can be seen that the quantum rotation gate is guided by the current optimal solution, and thus, it has a greater chance to converge to the optimal solutions.

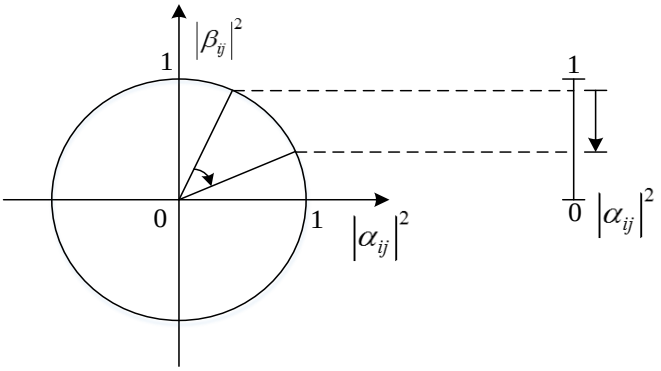


Fig. 2. Schematic diagram of the rotation gate

To further increase the diversity of the population, a mutation can be introduced as well. It can be seen as a quantum bit rotating 0.5π anticlockwise. This mutation process is identified by the users according to their preference.

3.4 A pruning strategy for the redundant itemsets

Because the updating process could produce some meaningless and redundant itemsets, a pruning strategy for such itemsets is designed in this section. It can be divided into three steps.

(1) Discovering all of the maximum mutually exclusive subsets in the studied transactions.

For the example in **Table 1**, we initialize the first maximum mutually exclusive subset by T_1, which is denoted as $S_1 = \{A, C, D\}$. When processing T_2, it can be seen that $T_2 \subseteq S_1$, and thus, the maximum mutually exclusive subset S_1 remains unchanged. For T_3, the maximum mutually exclusive subset S_1 is updated as $S_1 = \{A, C, D, E\}$ since $S_1 \subseteq T_3$. In contrast to T_2 and T_3, $T_4 \not\subseteq S_1$ and $S_1 \not\subseteq T_4$, and thus, the second maximum mutually exclusive subset S_2 is generated as $S_2 = \{B, C, D\}$. After handling the remaining transactions similar to the upper process, we can finally obtain that all of the maximum mutually exclusive subsets for the example in **Table 1** are $S_1 = \{A, C, D, E\}, S_2 = \{B, C, D\}, S_3 = \{C, F\}, S_4 = \{A, F\}$.

(2) Calculate the measured value of all of the chromosomes in the current population by the method below.

X_i is defined as a measured value of C_i correspondingly, which is represented by a binary string whose length is the number of 1-SHPUIs. The measurement process can be described as follows: generate a random number in (0,1), and if it is larger than $|\alpha_{ij}|^2$, X_{ij} is set to 1; otherwise, it is 0. Then, repeat this calculation for $sizepop \times n$ times. When the discovered 1-SHPUIs are sorted in alphabetic ascending order, if $X_{ij} = 1$, then the j-th 1-SHPUI is a member of the represented itemset; otherwise, it is not.

Because the number of 1-SHPUIs in **Table 5** is 5, the length of each chromosome is set as 5. When set $popsize$ as 6, one possible measured value of the population is shown in **Fig. 3**. The itemset that corresponds to X_1 is $\{AB\}$; for X_2, it is $\{AE\}$, and so on.

	A	*B*	*C*	*D*	*E*
X_1	1	1	0	0	0
X_2	1	0	0	0	1
X_3	0	1	1	0	1
X_4	1	0	1	0	0
X_5	0	1	0	0	1
X_6	0	0	1	1	1

Fig. 3. Example of possible measured values

(3) If the itemset that corresponds to the measured value X_i is not contained in these maximum mutually exclusive subsets, then X_i is deleted from the current population. Otherwise, its corresponding measured value remains unchanged in the current population.
For the possible measured values in **Fig. 3**, the itemset that corresponds to X_1 is $\{AB\}$, and the itemset that corresponds to X_2 is $\{AE\}$. Because $\{AB\}$ is not contained in $S_1 \sim S_4$, its corresponding measured value X_1 will be deleted from the current population. However, because $\{AE\}$ is contained in S_1, its corresponding measured value remains unchanged.

3.5 Fitness evaluation

The designed algorithm employs the potential utility of itemsets as the fitness function, which is denoted by

$$pu(X_r) = \sum_{X_i \subseteq T_q} pu(X_r, T_q) = \sum_{X_r \subseteq T_q} \sum_{i_p \subseteq X_r} pu(i_p, T_q) \tag{5}$$

and

$$pu(i_p, T_q) = iu(i_p, T_q) \times p(i_p, T_q) \times pu(i_p). \tag{6}$$

where X_r is the remaining measured value after adopting the pruning strategy. If the potential utility value of X_r is no smaller than the minimum utility threshold, it is considered to be a nHPUI and will be placed in the set of HPUIs.

3.6 Algorithm description and analysis

The above phases are then repeatedly processed until the termination condition is reached. The algorithm of the designed HPUIM-QGA is shown in **Table 7**.

Table 7. Description of the HPUIM-QGA algorithm.

Input:	An uncertain dataset *D*; a profit table *eu*; the minimum utility threshold δ; and the size of each population *M*.
Output:	A set of potential high utility itemsets *HPUIs*
1	For each $T_q \in D$ do
2	For each $i_j \subseteq T_q$ do
3	$ptu(T_q) = iu(i_j, T_q) \times p(i_j, T_q) \times eu(i_j)$;
4	End for
5	End for
6	Calculate $twpu(i_j) = \sum_{i_j \subseteq T_q} ptu(T_q)$;

7 $PTU = \sum_{T_q \in D} ptu(T_q)$;
8 Find 1-SHPUIs $\leftarrow \{i_j | twpu(i_j) \geq PTU \times \delta\}$;
9 Set $k = |1 - \text{SHPUIs}|$;
10 Initialize $C_i = \begin{bmatrix} \alpha_{i1} & \cdots & \alpha_{ik} \\ \beta_{i1} & \cdots & \beta_{ik} \end{bmatrix} = \begin{bmatrix} 1/\sqrt{2} & \cdots & 1/\sqrt{2} \\ 1/\sqrt{2} & \cdots & 1/\sqrt{2} \end{bmatrix}$;
11 While *termination criteria are not reached* do
12 For $i \leftarrow 1$ *to* M do
13 Calculate measured value X_i of C_i;
14 Pruning redundant itemsets by the designed strategy;
15 If $popsize > 0$ then
16 If $fitness(X_r) \geq PTU \times \delta$ then
17 $PHUIs \leftarrow GetItem(X_r) \cup PHUIs$
18 End if
19 End if
20 End for
21 Update quantum rotation gate according to the given strategy;
22 End while

In the designed HPUIM-QGA algorithm, the 1-SHPUIs are first discovered, and their number is set to be the length of the chromosomes in the evolutionary process (Lines 1 to 9). The chromosomes are encoded as quantum bits, and they are initialized as $1/\sqrt{2}$ (Line 10). Lines 11 to 22 are the evolutionary process, where the fitness evaluation phase, updating process and pruning strategy are executed until the termination criterion is reached. The function $GetItem(X_i)$ is used to convert chromosome X_i into its corresponding itemsets by the method in section 4.3. In the QGA, the termination criterion can usually be set to the maximum number of iterations.

4. Experimental Classification Results and Analysis

The proposed HPUIM-QGA algorithm considers both the tuple uncertainty and the attribute uncertainty. However, PHUI-List and PHUI-UP employ only the tuple uncertainty model, and thus, it is impossible to compare the HPUIM-QGA with PHUI-List and PHUI-UP directly. The same discussion can be applied to UHUI-Apriori but does not utilize the internal utility in the original definitions. Therefore, to prove that the proposed algorithm is acceptable, it is compared with HUITWU [10], which is a state-of-the-art algorithm for HUIM, and it can derive the complete condensed representation of the HUIs. Furthermore, to prove that the proposed algorithm is efficient, the proposed algorithm is also compared with HUPEumu-GRAM [11] and HUIM-BPSO [12]. Comparison experiments for these algorithms in terms of the runtime, memory consumption, and analysis for the discovered itemsets are conducted below.

All of the algorithms were implemented in Matlab, and the experiments were conducted on a personal computer equipped with an Intel Core i5-4590 dual-core processor and 4 GB of RAM, running the 32-bit Microsoft Windows 7 operation system. The experimental results are presented and discussed thereafter.

4.1 Datasets

Experiments were performed on the real-life datasets mushroom, connect and accidents, and the synthetic dataset T10I4D100K [20,21], which are widely used in the issue of HUIM. Parameters and characteristics of these datasets are respectively shown in **Tables 8** and **9**.

Table 8. Parameters of the used datasets.

# $\|D\|$	Total number of transactions
# $\|I\|$	Number of distinct items
AvgLen	Average transaction length
Type	Dataset type: sparse or dense

Table 9. Characteristics of the datasets.

Dataset	# $\|D\|$	# $\|I\|$	AvgLen	Type
T10I4D100K	100000	870	10.1	sparse
mushroom	8124	119	7.2	dense
connect	67557	132	43	dense
accidents	340183	468	33.8	dense

Because these datasets do not provide the external utility, internal utility and existence probability of each item, a simulation model [8] is employed. The model generates random numbers that obey the log-normal distribution in the [1, 5] interval and [1, 1000] interval, which correspond to the internal and external utility, respectively. In addition, due to the uncertainty property of the items in each transaction, their existence probabilities obey a uniform distribution in the [0.5, 1] interval. For HUITWU, HUPEumu-GRAM and HUIM-BPSO, precise versions of these datasets are employed since they only can address HUIM in precise datasets.

4.2 Runtime

In this paper, typical parameters are adopted in the compared algorithms. For the HUIM-BPSO algorithm, w_1 is set to 0.9, and the individual factor c_1 and the social factor c_2 are both set to 2, which is employed in the literature [12]. The parameters in the HUPEumu-GRAM algorithm originate from the literature [11], using roulette wheel selection, one-point crossover, and ranked mutation. The crossover rate is initially set to 0.9 in the experiment. For our proposed HPUIM-QGA, we adopt the selection strategy of rotation angle, as in section 4.4. When we set the number of iterations and population size to 10000 and 20 respectively, the runtime of the HPUIM-QGA, HUITWU, HUPEumu-GRAM and HUIM-BPSO in different minimum utility thresholds over the stated four datasets are shown in **Fig. 4**. Note that the minimum utility threshold is also used as the minimum potential utility threshold that is employed in the proposed HPUIM-QGA.

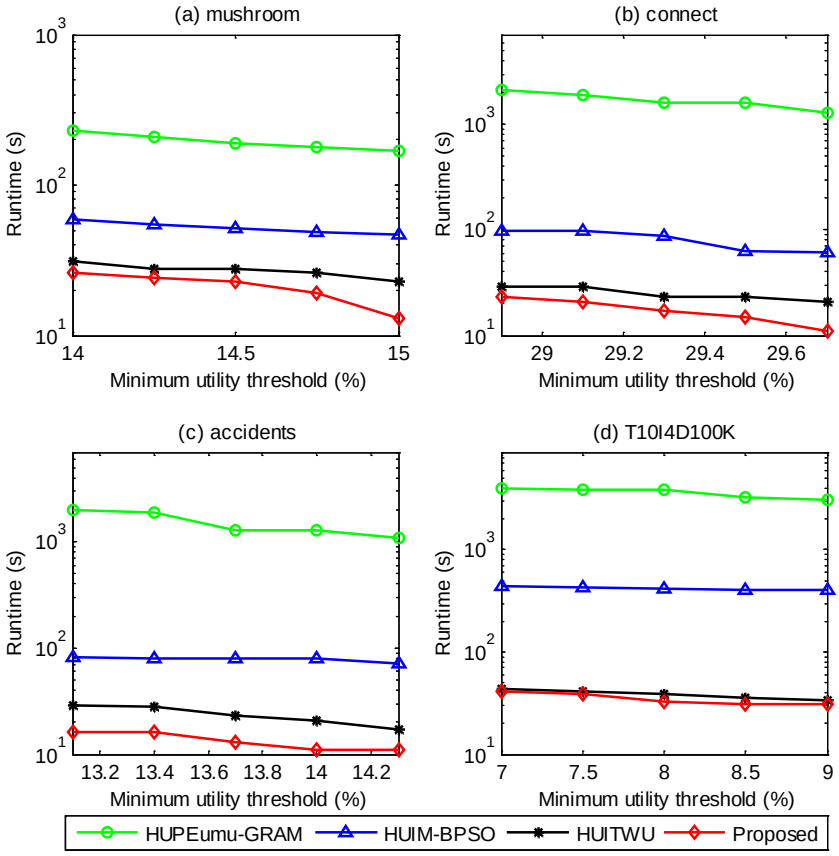


Fig. 4. Runtime under various values of δ

From **Fig. 4**, it can be seen that the runtime of the HUPEumu-GRAM, HUIM-BPSO, HUITWU, and HPUIM-QGA is in accordance with the order from long to short. HUPEumu-GRAM needs a long runtime since it requires us to initialize the chromosomes as the HUIs in the evolutionary process, which consumes time to find these initial chromosomes from the large number of unpromising and meaningless itemsets. While the quantum chromosome can be used to characterize multiple states simultaneously, which implies strong parallelism and better ability to maintain the diversity of population, the run time of the HPUIM-QGA is shorter than that of HUIM-BPSO, while a pruning strategy is adopted in these two algorithms. Evolutionary computation that can find the optimal solutions quickly in the dense datasets and the construction of the HUITWU-Tree that is developed in the HUITWU consumes time, and thus, the runtime of the HPUIM-QGA is shorter than that of HUITWU in the dense datasets such as mushroom, connect and accidents. However, a large number of itemsets are not useful in the sparse dataset T10I4D100K, which makes the time consumption for the pruning strategy and updating process increase. Therefore, the runtime of the HPUIM-QGA is very close to that of HUITWU. Moreover, from **Fig. 4**, it can be seen that the runtime is inversely proportional to the minimum utility threshold δ in general. This phenomenon arises because the larger the value of δ is, the smaller the number of HUIs or HPUIs.

4.3 Memory consumption

Using the same parameter settings that are adopted in the section 4.2, the memory consumption is evaluated to show the performance of the compared four algorithms. The comparison results are shown in **Fig. 5**.

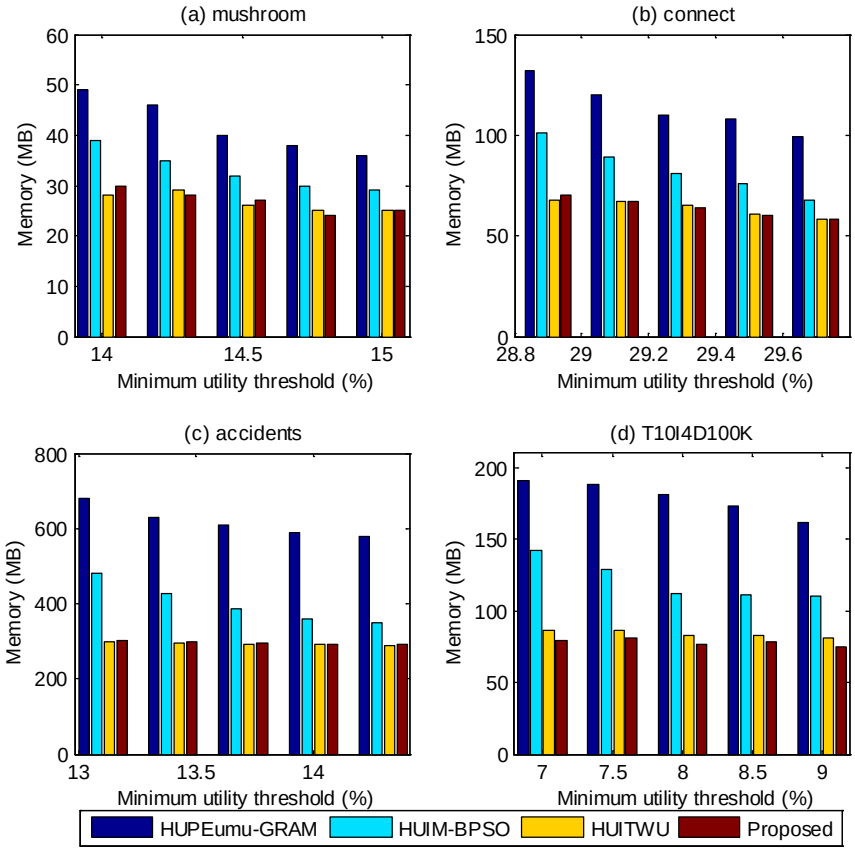


Fig. 5. Memory consumption under various values of δ

From **Fig. 5**, it can be seen that the memory consumption is also inversely proportional to the minimum utility threshold δ in general. This result arises because all of these four compared algorithms adopt the overestimation of the utility or potential utility, which prunes the unpromising itemsets in an early stage by using its downward property. Therefore, the larger the value of δ is, the smaller the values of 1-SHPUIs or 1-HTWUIs (high transaction weighted itemsets). Furthermore, because the number of 1-SHPUIs or 1-HTWUIs is a decided factor for the length of chromosome and the scale of HUITWU-Tree, the smaller the numbers of 1-SHPUIs or 1-HTWUIs are, the lower the amount of memory consumption. It is easy to understand that the memory consumption largely depends on the popsize for the evolutionary algorithms. Therefore, compared to HUPEumu-GRAM and HUIM-BPSO, our proposed HPUIM-QGA consumes less memory since the redundant itemsets are pruned from the current population, which leads to the popsize for the HPUIM-QGA becoming small. As HUPEumu-GRAM needs additional calculations to initialize the chromosomes as HUIs, the memory consumption of HUIM-BPSO is less than that of HUPEumu-GRAM, while their popsize is the same. The experimental results in **Fig. 5** show that the memory consumption for

the HUITWU-Tree is less than that of HUIM-BPSO. The reason for this result is that only pruning or inserting operations are needed to maintain the HUITWU-Tree when the minimum utility threshold δ changes, while HUIM-BPSO needs an updating operation in every iteration. From **Fig. 5**, it can also be seen that the memory consumption of the HPUIM-QGA is close to that of HUITWU in the dense datasets such as mushroom, connect and accidents. However, a large number of itemsets are not useful in the sparse dataset T10I4D100K, which makes the popsize for the HPUIM-QGA become smaller. Therefore, the memory consumption of the HPUIM-QGA is less than that of HUITWU.

4.4 Analysis of the discovered itemsets

Using the same parameter settings that are adopted in section 5.2, the number of HUIs and HPUIs are compared in this section to evaluate whether the proposed algorithm can be accepted. To prove that the proposed HPUIM-QGA is reasonable, the precise version of it, called the HUIM-QGA, is conducted in this section by setting all of the existence probabilities of the items in the uncertain datasets to 1. **Fig. 6** shows the number of discovered itemsets by the HUPEumu-GRAM, HUIM-BPSO, HUITWU, HUIM-QGA and the proposed HPUIM-QGA under various δ. Moreover, **Table 10** evaluates the compression ratio of the HPUIs that are generated by the proposed algorithm.

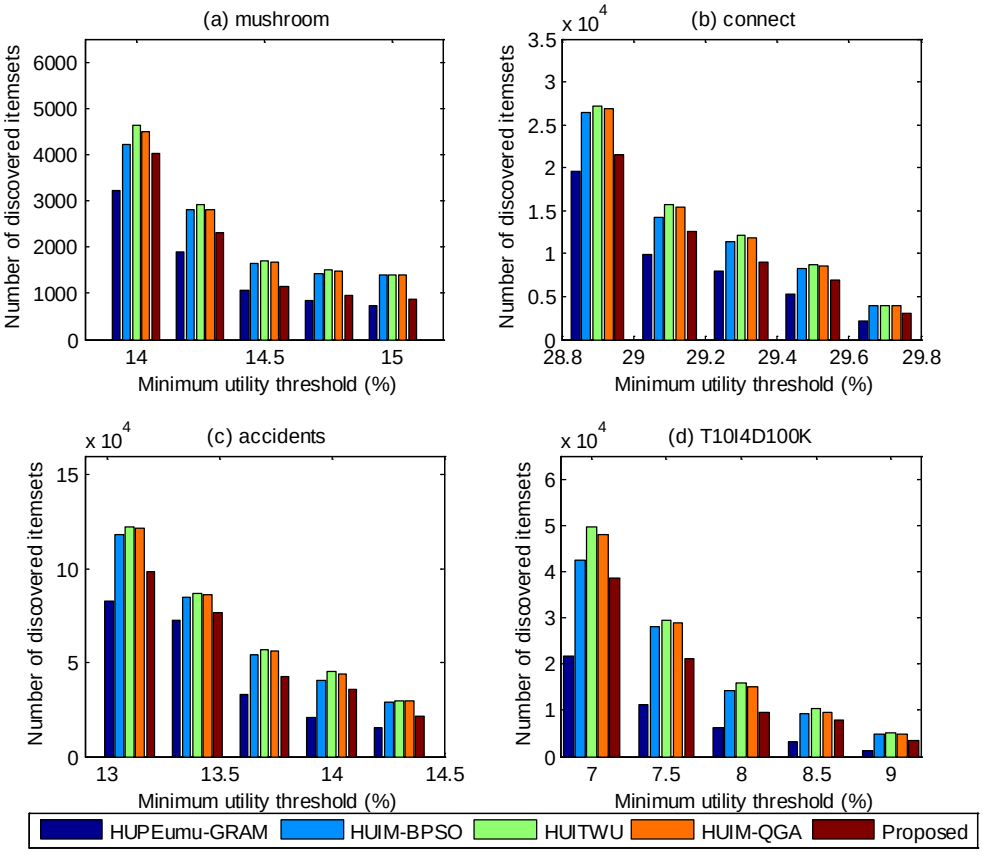


Fig. 6. Number of discovered itemsets under various values of δ

From **Fig. 6**, it can be seen that the number of HUIs discovered by the HUIM-QGA is close to that of HUITWU, which can discover the actual and complete HUIs from the precise datasets. This result demonstrates that our proposed algorithm has a strong ability to search out the HUIs and the mechanism of it is reasonable. In the current parameter settings, the number of HUIs discovered by HUIM-BPSO is smaller than that of the HUIM-QGA since the QGA has a higher convergence speed than that of BPSO. The experimental results in **Fig. 6** show that the number of HUIs discovered by HUPEumu-GRAM is much smaller than that of the HUIM-BPSO, HUITWU and HUIM-QGA, especially for the sparse dataset T10I4D100K. This result arises because HUPEumu-GRAM has no pruning strategy for the redundant itemsets, which leads to having a large number of calculations being useless and having the discovered itemsets being meaningless. Additionally from **Fig. 6**, for various δ on the four datasets, it can be observed that the number of HUIs is larger than that of HPUIs, which are discovered by our proposed HPUIM-QGA. This finding arises because the proposed algorithm considers both the existence probability and the utility, while HUITWU considers only the utility. This result reflects that fewer HPUIs are produced from the numerous discovered HUIs when considering the existence probability constraint. Therefore, in real-life applications, a large number of HUIs might not be the itemsets needed by the users for making efficient decisions because the HUIs with a high existence probability are useful to them, similar to HPUIs, especially when the δ is set to be low. From **Fig. 6**, it can also be seen that the number of HUIs and HPUIs is inversely proportional to δ. The reason is that many unpromising candidate itemsets are pruned when the δ value is set high, and thus, the algorithms can avoid handling them in the mining process.

Furthermore, the compression ratio of the HPUIs is studied in this paper, and it is defined as follows.

$$CRHEUIs = \frac{|HUIs - HPUIs|}{|HUIs|}$$

Comparison results are shown in **Table 10**.

Table 10. Analysis of $CRHPUIs$ under various δ.

mushroom	14%	14.25%	14.5%	14.75%	15%
HUIs	4639	2923	1712	1496	1395
HPUIs	4016	2322	1156	968	869
$CRHEUIs$	13.43%	20.56%	32.48%	35.29%	37.71%
connect	28.9%	29.1%	29.3%	29.5%	29.7%
HUIs	27123	15365	12135	8652	3966
HPUIs	21569	12545	8965	6852	3126
$CRHEUIs$	20.48%	18.35%	26.12%	20.80%	21.18%
accidents	13.1%	13.4%	13.7%	14%	14.3%
HUIs	122536	86598	56897	45235	29856
HPUIs	98653	76564	42567	35641	21358
$CRHEUIs$	19.49%	11.59%	25.19%	21.21%	28.46%
T10I4D100K	7%	7.5%	8%	8.5%	9%
HUIs	49652	29562	15896	10256	4988
HPUIs	38564	21214	9634	7985	3541
$CRHEUIs$	22.33%	28.24%	39.39%	22.14%	29.01%

From **Table 10**, it can be seen that the compression ratio achieved by mining HPUIs is relatively high. This finding means that many meaningless itemsets are pruned by considering both the existence probability and the utility. From **Fig. 6** and **Table 10**, we can conclude that the proposed algorithm can discover itemsets that have high utility and high probability over uncertain datasets, which demonstrates that the proposed HPUIM-QGA is reasonable and acceptable.

4.5 Convergence

Because increases in the maximum number of iterations and the popsize can lead to increases in the runtime and memory consumption, the proper settings can make the proposed algorithm efficient. The experiments in this section are conducted to prove that our proposed algorithm can converge to the optimal chromosome quickly and that it is reasonable to set the maximum iteration and popsize to 10000 and 20, respectively. Additionally, the precise version of our proposed algorithm, which is called the HUIM-QGA and is defined in section 5.4, is used in this section since it can reflect the performance of our proposed HPUIM-QGA, and its results can be compared to HUPEumu-GRAM, HUIM-BPSO and HUITWU directly.

First, we set the popsize to 20, and given the minimum utility threshold δ (the percentages that are presented in the subtitles), the convergence of the HUPEumu-GRAM, HUIM-BPSO and HUIM-QGA are evaluated under various iterations through comparing with the actual and complete HUIs discovered by HUITWU. The comparison results are shown in **Fig. 7**.

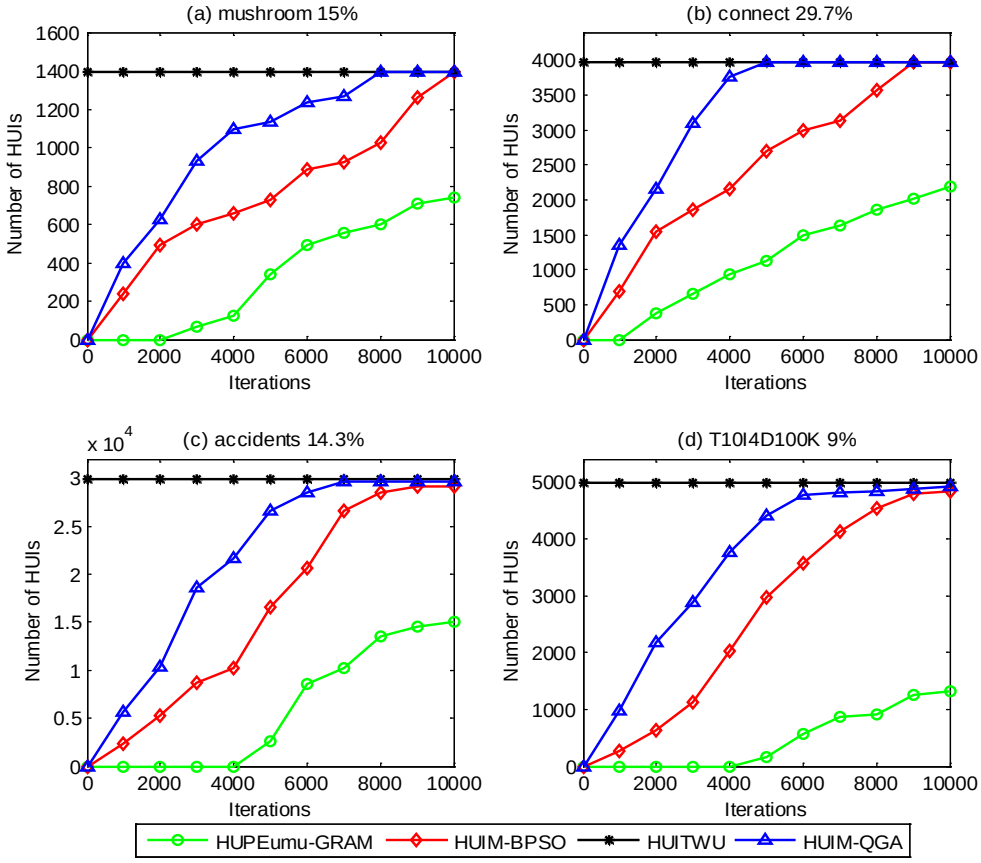


Fig. 7. Convergence under various iterations when *popsize*=20

From **Fig. 7**, it can be seen that the convergence speed of the HUIM-QGA is quicker than that of HUPEumu-GRAM and HUIM-BPSO. For dense datasets, the HUIM-QGA can even converge to the number of HUIs that are discovered by HUITWU at the 10000-th iteration. A small difference occurred in the sparse dataset T10I4D100K since the number of HUIs is very small, but this difference is in the tolerance.

Second, we set the maximum iteration to 1000, and given the minimum utility threshold δ (the percentages that are presented in the subtitles), the convergence of the HUPEumu-GRAM, HUIM-BPSO and HUIM-QGA are evaluated under various values of popsize through comparing the actual and complete HUIs discovered by HUITWU. The comparison results are shown in **Fig. 8**.

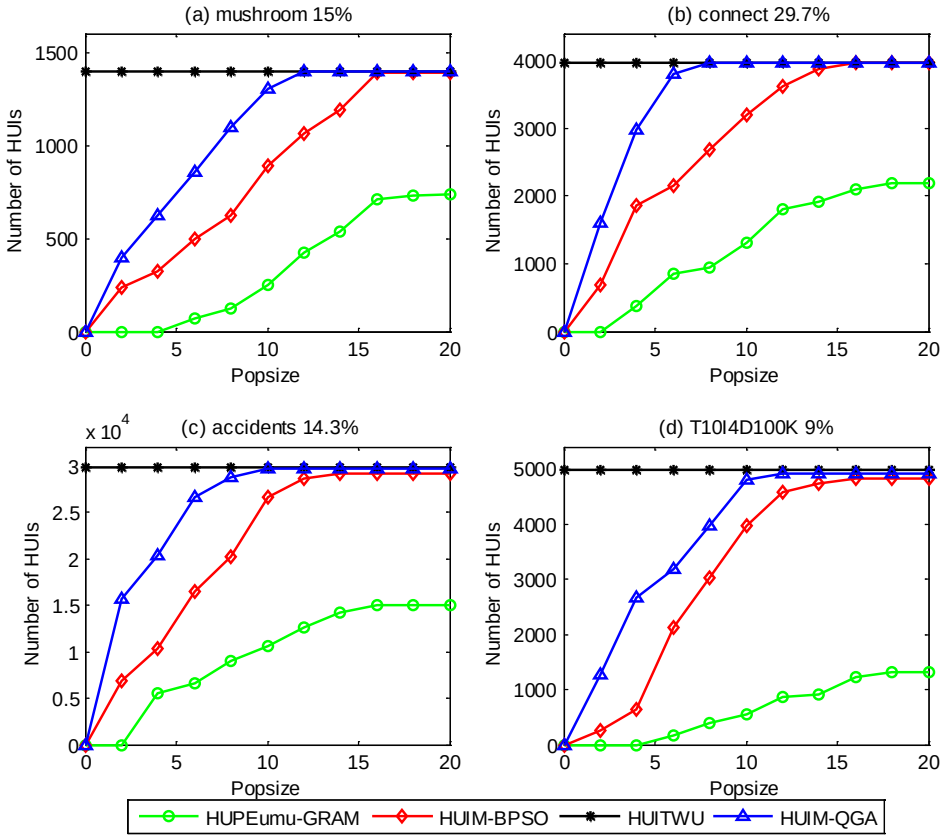


Fig. 8. Convergence under various *popsize* when maximum iteration=10000

From **Fig. 8**, it can also be seen that the convergence speed of the HUIM-QGA is quicker than that of HUPEumu-GRAM and HUIM-BPSO, and the HUIM-QGA can converge to the number of HUIs discovered by HUITWU when the popsize is set to 20.

Both these two experiments prove that the proposed algorithm can converge to the optimal chromosome quickly. These results also show that the HUIM-QGA achieves good convergence when the maximum iteration and popsize are set to 10000 and 20, respectively. These two experiments provide a method to determine the settings of the maximum number of iterations and the popsize. In other words, through various iterations and popsize, when the convergence of the HUIM-QGA satisfies the requirements of the users, the corresponding settings can be employed in the subsequent experiments.

5. Conclusions

In the precise datasets, many algorithms have been proposed to mine HUIs, where most of them employ the framework of Apriori and FP-tree. Moreover, HUPEumu-GRAM and HUIM-BPSO are the only two algorithms based on evolutionary computation that address the problem of HUIM.

In the uncertain datasets, the itemsets that have a high utility and high existence probability are useful to the users, not the itemsets with only one of these properties. To the best of our knowledge, Lin et al. proposed PHUI-UP based on a two-phase model and PHUI-List based on a list structure, while Lan et al. proposed UHUI-apriori based on Apriori, and these are the only algorithms that are used to solve the HUIM problem over uncertain datasets. In other words, HUIM over uncertain datasets is a relatively new issue, and new algorithms based on evolutionary algorithms are waiting to be developed.

According to the above research, new definitions of items utility, itemsets utility, transaction utility and transaction weighted utility are given. Additionally, an algorithm of HUIM over uncertain datasets is proposed based on the QGA in this paper, which treats the number of 1-SHPUIs as the size of the chromosomes. It can not only cut down the search space by UBPU and the designed pruning strategy but also utilize the QGA's advantage in handling the combinatorial explosion problem. Experiments on real-life and synthetic uncertain datasets show that the proposed algorithm has good performance in terms of the runtime, memory consumption, analysis for the discovered itemsets and convergence.

To the best of our knowledge, this paper shows the first algorithm that involves discovering HPUIs based on evolutionary computation over uncertain datasets. The QGA-based approach requires very few parameters to be set and has advantages in searching capability, convergence, computing time, and so on. Thus, these characteristics determine that the proposed algorithm can have good performance on HUIM over uncertain datasets. Of course, other evolutionary algorithms can be used in this problem also, but it is a non-trivial task since some algorithms can only be used to handle the continuous problem and some algorithms perform the operations with randomization, which cannot be applied to address this issue directly. More work can be performed in applying evolutionary algorithms to HUIM over uncertain datasets, and more mechanisms can be studied in the near future.

References

[1] R. Agrawal, T. Imielinski, A. Swami, "Database mining: a performance perspective," *IEEE Transactions on Knowledge and Data Engineering*, vol. 5, no. 6, pp. 914-925, December, 1993. Article (CrossRef Link)

[2] R. Agrawal, T. Imielinski, A. Swami, "Mining association rules between sets of items in large dataset," in *Proc. of the 1993 ACM SIGMOD International Conference on Management of Data*, pp. 207-216, May 25-28, 1993. Article (CrossRef Link)

[3] R. Agrawal, R. Srikant, "Fast algorithms for mining association rules," in *Proc. of the 20th international conference on very large datasets*, pp. 487-499, September 12-15, 1994. Article (CrossRef Link)

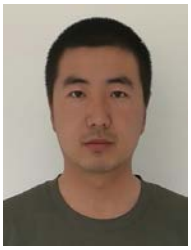
[4] J. Han, J. Pei, Y. Yin, R. Mao, "Mining frequent patterns without candidate generation: a frequent-pattern tree approach," *Data Mining and Knowledge Discovery*, vol. 8, no. 1, pp. 53-87, February, 2004. Article (CrossRef Link)

[5] H. Yao, H.J. Hamilton, C.J. Butz, "A foundational approach to mining itemset utilities from datasets," in *Proc. of the 2004 SIAM International Conference on Data Mining*, pp. 482-486, April 22-24, 2004. Article (CrossRef Link)

[6] Z. Morteza, A. Aijun, "Mining top-k high utility patterns over data streams," *Information Science*, vol. 285, no. 1, pp. 138-161, January, 2014. Article (CrossRef Link)
[7] R. Chan, Q. Yang, Y.D. Shen, "Mining high utility itemsets," in *Proc. of IEEE International Conference on Data Mining*, pp.19-26, November, 2003. Article (CrossRef Link)
[8] Y. Liu, W.K. Liao, A. Choudhary, "A two-phase algorithm for fast discovery of high utility itemsets," in *Proc. of the 9th Pacific-Asia conference on Advances in Knowledge Discovery and Data Mining*, pp. 689-695, May 18-20, 2005.Article (CrossRef Link)
[9] C.W. Lin, T.P. Hong, W.H. Lu, "An effective tree structure for mining high utility itemsets," *Expert Systems with Application*, vol. 38, no. 6, pp. 7419-7424, June, 2011. Article (CrossRef Link)
[10] G. Shiming, G. Hong. "HUITWU: An Efficient Algorithm for High-Utility Itemset Mining in Transaction Databases," *Joural of Computer Science and Technology*, vol. 31, no. 4, pp. 776-786, July, 2016. Article (CrossRef Link)
[11] S. Kannimuthu, K. Premalatha, "Discovery of high utility itemsets using genetic algorithm with ranked mutation," *Applied Artificial Intelligence*, vol. 28, no. 4, pp. 337-359, April, 2014. Article (CrossRef Link)
[12] C. L. Lin, Y. Lu, F.V. Philippe, T.P. Hong, V. Miroslav, "A binary PSO approach to mine high-utility itemsets," *Soft Comput*, vol. 21, no. 17, pp. 5103-5121, March, 2016. Article (CrossRef Link)
[13] J.C. Lin, W. Gan, F.V. Philippe, T.P. Hong, V.S. Tseng, "Efficient algorithms for mining high-utility itemsets in uncertain datasets," *Knowledge-Based Systems*, vol. 96, no. C, pp. 171-187, March, 2016. Article (CrossRef Link)
[14] Y.Q. Lan, Y. Wang, Y. Wang, S.W. Yi, D. Yu, "Mining high utility itemsets over uncertain datasets," in *Proc. of International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*, pp. 235-238, September 17-19, 2015. Article (CrossRef Link)
[15] H. Huan, C. Liang, W. Feng-ge, W. Hua-li, W. Yaqi. "An Improved Dynamic Quantum Genetic Algorithm," *Journal of Military Communications Technology*, vol. 38, no. 2, pp. 17-21, February, 2017. Article (CrossRef Link)
[16] H. Liuyu, G. Shuping, W. Junning, X. Xiaona. "Hybrid frog leaping algorithm based on differential evolution," *Systems Engineering and Electronics*, vol. 39, no. 10, pp. 2382-2391, October, 2017. Article (CrossRef Link)
[17] L. Dongsheng, G. Yang, Y. Aixia. "Jamming Resource Allocation via Improved Discrete Cuckoo Search Algorithm," *Journal of Electronics & Information Technology*, vol. 38, no. 4, pp. 899-905, April, 2016. Article (CrossRef Link)
[18] C. Hao, Z. Jie, Y. Qingping, D. Yaya, X. Lixue, J. Minjie. "Multi-population artificial bee colony algorithm based on hybrid search," *Journal of Computer Applications*, vol. 37, no. 10, pp. 2773-2779, October, 2017. Article (CrossRef Link)
[19] C.K. Chui, B. Kao, E. Hung. "Mining frequent itemsets from uncertain data," in *Proc. of the Pacific-Asia conference advances in knowledge discovery and data mining*, pp. 47-58, May 22-25, 2007. Article (CrossRef Link)
[20] J. Tang, A. Liu, M. Zhao, T. Wang. "An Aggregate Signature based Trust Routing for Data Gathering in Sensor Networks," *Security and Communication Networks*, vol. 2018, January, 2018. Article (CrossRef Link)
[21] Y. Liu, A. Liu, S. Guo, Z. Li, Y.J. Choi. "Contest-aware collect data with energy efficient in Cyber-physical cloud systems," *Future Generation Computer Systems*, available online, June, 2017. Article (CrossRef Link)

Ju Wang received the B.S. and M.S. degree in 2012 and 2014 respectively, from Air Force Engineering University, Xi'an, China. She is working towards the Ph.D. degree in the Air Force Engineering University. Her research interests include data mining and evolutionary algorithms.

Fuxian Liu is professor and PhD supervisor of Air Force Engineering University. His main research interests include decision analysis and data mining.



Chunjie Jin received the B.S. degree in 2012, from Hebei University of Technology, Tianjin, China. He is an assistant engineer now. His research interests include data mining and evolutionary algorithms.