Mining Quantitative Correlated Patterns Using an Information-Theoretic Approach[*]

Yiping Ke
keyiping@cse.ust.hk

James Cheng
csjames@cse.ust.hk

Wilfred Ng
wilfred@cse.ust.hk

Department of Computer Science and Engineering
Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong

ABSTRACT

Existing research on mining quantitative databases mainly focuses on mining associations. However, mining associations is too expensive to be practical in many cases. In this paper, we study mining correlations from quantitative databases and show that it is a more effective approach than mining associations. We propose a new notion of Quantitative Correlated Patterns (QCPs), which is founded on two formal concepts, mutual information and all-confidence. We first devise a normalization on mutual information and apply it to QCP mining to capture the dependency between the attributes. We further adopt all-confidence as a quality measure to control, at a finer granularity, the dependency between the attributes with specific quantitative intervals. We also propose a supervised method to combine the consecutive intervals of the quantitative attributes based on mutual information, such that the interval combining is guided by the dependency between the attributes. We develop an algorithm, *QCoMine*, to efficiently mine QCPs by utilizing normalized mutual information and all-confidence to perform a two-level pruning. Our experiments verify the efficiency of *QCoMine* and the quality of the QCPs.

Categories and Subject Descriptors: H.2.8 [Database Management]: Database Applications - Data Mining

General Terms: Algorithms

Keywords: Quantitative Databases, Correlated Patterns, Information-Theoretic Approach, Mutual Information

1. INTRODUCTION

Mining correlations [3, 5, 11, 10, 18, 9] is recognized as an important data mining task for its many advantages over mining association rules [1]. Instead of discovering co-occurrence patterns in data, mining correlations identifies the underlying dependency between the attributes in a pattern. More importantly, mining correlations does not rely on the *support* measure to perform pruning; thus, correlated patterns are not restricted to frequently co-occurring attributes, and those infrequent but significant patterns that are too expensive to be obtained by association rule mining can also be discovered. This property of correlation is very useful for the discovery of rarely occurring but important incidents, such as diseases, network intrusions, earthquakes and so on, and their possible causes.

Existing research on mining correlations is primarily conducted on boolean databases. However, most attributes in real-life databases can be *quantitative*, which are numeric values (e.g. salary), and *categorical*, which are enumerations (e.g. education level). We refer to these databases as *quantitative databases*. A boolean database is in fact a special quantitative database that only has categorical attributes with boolean values. Thus, mining quantitative databases is a more general problem in its own right but a harder problem from the technical perspective than mining boolean databases.

In this paper, we propose to mine correlations from quantitative databases using an information-theoretic approach. We study the properties of *Mutual Information* (*MI*) [6] on quantitative databases and define *Normalized Mutual Information* (*NMI*) that is to be applied in the context of correlation mining. Then, we propose a new notion of *Quantitative Correlated Patterns* (*QCPs*) based on NMI and the well-established correlation measure, *all-confidence* [12, 11]. This new definition of QCPs achieves two levels of quality control on the mining result. First, we employ NMI to specify a required minimum degree of dependency among all attributes in a pattern. Then, we use all-confidence to enforce correlation at a finer granularity on the specific intervals of the quantitative attributes.

The first step in mining quantitative databases is to discretize the large domain of a quantitative attribute into a number of small intervals. During the mining process, consecutive intervals of an attribute may need to be combined to gain sufficient support value as well as to produce meaningful intervals [15, 16]. We develop a *supervised interval combining method* specifically for correlation mining so that the combined intervals also capture the dependency between the attributes, thereby ensuring the quality of the mined correlations. Our interval combining method utilizes MI to guide the interval combining of one attribute with respect to another attribute. We model the interval combining prob-

[*]This work is partially supported by RGC CERG under grant number HKUST6185/03E.

lem as an optimization problem and devise a fast greedy algorithm as a solution.

We develop an efficient algorithm, *QCoMine*, for mining QCPs. The algorithm is built on two effective pruning techniques: the *attribute-level pruning* by NMI and the *interval-level pruning* by all-confidence. First, at the attribute level, we define an *NMI graph* on all attributes such that an edge exists between two attributes only if their NMI exceeds a pre-defined threshold. We incorporate the NMI graph into our mining process, which effectively prunes an overwhelming number of uncorrelated patterns that are generated from those attributes with low mutual dependency. Then, at the interval level, all-confidence is applied to further prune the uncorrelated intervals of the highly dependent attributes. With its downward closure property, all-confidence is able to quickly converge a large search space to a small and promising one.

Our experiments show that the supervised interval combining method and the pruning by NMI and all-confidence are the keys to efficient correlation mining from quantitative databases. Without any one of them, *QCoMine* either uses substantially more resources (orders of magnitude greater running time and memory usage) or is unable to complete the mining (exhausting the memory). The patterns mined by *QCoMine* not only reveal the effectiveness of our interval combining method in obtaining meaningful intervals for correlated attributes, but also verify the efficiency of NMI and all-confidence in pruning uncorrelated patterns. We further examine the feasibility of mining frequent patterns from quantitative databases [15], compared with our approach of mining correlated patterns. We find that frequent patterns are mostly patterns with either very low all-confidence (i.e., uncorrelated) or trivial intervals (i.e., common-knowledge), while the majority of the patterns obtained by *QCoMine* are rare but highly correlated. When quantitative frequent pattern mining becomes infeasible even under very restrictive settings such as very large minimum support thresholds, *QCoMine* still achieves an impressive performance.

Organization. We give preliminaries in Section 2. We define NMI in Section 3, based on which we propose a new notion of QCPs in Section 4. We present our supervised interval combining method in Section 5 and our mining algorithm, *QCoMine*, in Section 6. Then, we analyze the performance study in Section 7. Finally, we discuss related work in Section 8 and conclude our paper in Section 9.

2. PRELIMINARIES

Let $\mathcal{I} = \{x_1, x_2, \ldots, x_m\}$ be a set of distinct *attributes* or *random variables*[1]. These attributes can either be *categorical* or *quantitative*. Let $dom(x_j)$ be the domain of an attribute x_j, for $1 \leq j \leq m$. An *item*, denoted as $x[l_x, u_x]$, is an attribute x associated with an *interval* $[l_x, u_x]$, where $x \in \mathcal{I}$ and $l_x, u_x \in dom(x)$. We have $l_x = u_x$ if x is categorical and $l_x \leq u_x$ if x is quantitative. A *quantitative pattern* (or simply called *pattern*) is a nonempty set of items with distinct attributes. Given a pattern X, we define its attribute set as $attr(X) = \{x \mid x[l_x, u_x] \in X\}$ and its interval set as $interval(X) = \{[l_x, u_x] \mid x[l_x, u_x] \in X\}$. A pattern X is called a k-pattern if $|attr(X)| = k$. Similarly, we define k-attribute set and k-interval set, where k is the cardinality of the respective set. Given two patterns X and Y, we say X is a *sub-pattern* of Y (or Y is a *super-pattern* of X) if $\forall x[u_x, l_x] \in X$, we have $x[u_x, l_x] \in Y$. For brevity, we write a pattern $X = \{x[l_x, u_x], y[l_y, u_y]\}$ as $x[l_x, u_x]y[l_y, u_y]$.

[1]We use the terms *attribute* and *random variable* interchangeably in subsequent discussions.

For simplicity of discussion, we assume a lexicographic order in the set of attributes $\mathcal{I}$. Thus, the items in a pattern are ordered according to the order of their attributes in $\mathcal{I}$.

A *transaction* T is a vector $\langle v_1, v_2, \ldots, v_m \rangle$, where $v_j \in dom(x_j)$, for $1 \leq j \leq m$. We say T *supports* a pattern X if $\forall x_k[l_k, u_k] \in X$, $l_k \leq v_k \leq u_k$, where $k \in \{1, \ldots, m\}$. A *quantitative database* $\mathcal{D}$ is a set of transactions. The *frequency* of a pattern X in $\mathcal{D}$, denoted by $freq(X)$, is the number of transactions in $\mathcal{D}$ that support X. The *support* of X, denoted by $supp(X)$, is the probability that a transaction T in $\mathcal{D}$ supports X, and is defined as $supp(X) = freq(X)/|\mathcal{D}|$.

Running Example Table 1 shows an employee database as a running example throughout the paper for illustration purpose. The database consists of six attributes: `age`, `education`, `gender`, `married`, `salary` and `service years`. The quantitative attributes include `age`, `salary` and `service years`. All the attributes are labelled with a set of consecutive integers. The last column of the table records the support value of each transaction value. An example pattern is $X = \texttt{age}[4,5]\texttt{gender}[1,1]$ and $supp(X) = 0.25 + 0.19 = 0.44$.

Table 1: Employee Database

age	education	gender	married	salary	service years	*supp*()
4	2	1	1	1	4	0.25
5	1	1	1	2	3	0.19
2	1	1	1	1	3	0.11
1	2	1	2	2	1	0.09
2	1	1	1	1	1	0.09
3	1	1	1	2	3	0.09
2	2	1	1	2	1	0.08
4	3	2	1	4	3	0.06
3	3	2	1	4	2	0.03
1	2	2	2	3	2	0.01

3. NORMALIZED MUTUAL INFORMATION

In this section, we first review the concepts of entropy and mutual information. Then, we propose a normalization of mutual information to make it applicable in mining correlations from quantitative databases.

Entropy and *Mutual Information* (*MI*) are two central concepts in information theory [6]. Entropy measures the uncertainty of a random variable, while MI describes how much information one random variable tells about another one.

Table 2: Notations

$x, y, \cdots$	random variables (or attributes)
v_x	the value of x in $dom(x)$
$p(v_x)$	the probability of $(x = v_x)$
$p(v_x, v_y)$	the joint probability of $(x = v_x)$ and $(y = v_y)$
$p(v_y \mid v_x)$	the conditional probability of $(y = v_y)$ given that $(x = v_x)$

Table 2 lists some notations used throughout this paper. In the context of mining quantitative databases, we have $p(v_x) = supp(x[v_x, v_x])$ and $p(v_x, v_y) = supp(x[v_x, v_x]y[v_y, v_y])$.

Definition 1 (Entropy) The *entropy* of a random variable x, denoted as $H(x)$, is defined as

$$H(x) = -\sum_{v_x \in dom(x)} p(v_x) \cdot \log p(v_x).$$

The *conditional entropy* of a random variable y given another variable x, denoted as $H(y|x)$, is defined as

$$H(y|x) = -\sum_{v_x \in dom(x)} \sum_{v_y \in dom(y)} p(v_x, v_y) \cdot \log p(v_y|v_x).$$

The *joint entropy* of two random variables x and y, denoted as $H(x,y)$, is defined as

$$H(x,y) = -\sum_{v_x \in dom(x)} \sum_{v_y \in dom(y)} p(v_x, v_y) \cdot \log p(v_x, v_y).$$

Definition 2 (Mutual Information) The *Mutual Information* (*MI*) of two random variables x and y, denoted as $I(x;y)$, is defined as

$$I(x;y) = \sum_{v_x \in dom(x)} \sum_{v_y \in dom(y)} p(v_x, v_y) \cdot \log \frac{p(v_x, v_y)}{p(v_x) \cdot p(v_y)}.$$

We now present some properties of MI that are used to develop a normalization on MI. Detailed proof can be found in [6].

Property 1 $I(x;y) = H(x) - H(x|y) = H(y) - H(y|x)$.

Property 1 gives an important interpretation of MI. The information that y tells us about x is the reduction in the uncertainty of x given the knowledge of y, and similarly for the information that x tells about y. The greater the value of $I(x;y)$, the more information x and y tell about each other.

Property 2 $I(x;y) = I(y;x)$.

Property 2 suggests that MI is *symmetric*, which means the amount of information x tells about y is the same as that y tells about x.

Property 3 $I(x;x) = H(x)$.

Property 3 states that the MI of x with itself is the entropy of x. Thus, entropy is also called *self-information*.

Property 4 $I(x;y) \geq 0$.

Property 4 gives the lower bound for MI. When $I(x;y) = 0$, we have $p(v_x, v_y) = p(v_x)p(v_y)$ for every possible values of v_x and v_y, which means that x and y are independent, that is, x and y tell us nothing about each other.

Property 5 $I(x;y) \leq H(x)$ and $I(x;y) \leq H(y)$.

Property 5 gives the upper bound for MI.

Property 6 $I(x;y) = H(x) + H(y) - H(x,y)$.

Property 6 shows that the MI of x and y is the uncertainty of x plus the uncertainty of y minus the uncertainty of both x and y.

Although MI serves as a good measure to quantify how closely two attributes are related to each other, the scale of the MI values does not fall in the unit range as shown by Properties 4 and 5. Property 5 indicates that the MI of two attributes is bounded by the minimum of their entropy. Since the entropy of different attributes varies greatly, the value of MI also varies for different pairs of attributes. To apply MI to our mining problem, we require a unified scale for measuring MI among a global set of attributes. For this purpose, we propose normalized MI as follows.

Definition 3 (Normalized Mutual Information) The *Normalized Mutual Information* (*NMI*) of two random variables x and y, denoted as $\widetilde{I}(x;y)$, is defined as

$$\widetilde{I}(x;y) = \frac{I(x;y)}{MAX\{I(x;x), I(y;y)\}}.$$

Our idea is to normalize the MI of x and y by the maximum MI of x (or y) and any other attribute in $\mathcal{I}$, which is either $I(x;x) = H(x)$ or $I(y;y) = H(y)$ as shown by Property 5. As a result, we eliminate the localness and make NMI a global measure. We now present some useful properties of NMI as follows.

Property 7 $\widetilde{I}(x;y) = \widetilde{I}(y;x)$.

Proof. It follows directly from Property 2. □

Property 7 shows that, the same as MI, NMI is also symmetric.

Property 8 $0 \leq \widetilde{I}(x;y) \leq 1$.

Proof. Since $I(x;x) \geq 0$, $I(y;y) \geq 0$ and $I(x;y) \geq 0$, we have $\widetilde{I}(x;y) \geq 0$. By Properties 3 and 5, $I(x;y) \leq MIN\{H(x), H(y)\} \leq MAX\{H(x), H(y)\} = MAX\{I(x;x), I(y;y)\}$, thus $\widetilde{I}(x;y) \leq 1$. □

This property ensures that the value of NMI falls within the unit range $[0,1]$.

Property 9 $\widetilde{I}(x;y) = MIN\{\frac{H(x)-H(x|y)}{H(x)}, \frac{H(y)-H(y|x)}{H(y)}\}$.

Proof. By Properties 1 and 3, we have $\widetilde{I}(x;y) = MIN\{\frac{I(x;y)}{I(x;x)}, \frac{I(x;y)}{I(y;y)}\} = MIN\{\frac{H(x)-H(x|y)}{H(x)}, \frac{H(y)-H(y|x)}{H(y)}\}$. □

Property 9 gives the semantics of NMI, that is *the minimum percentage of reduction in the uncertainty of one attribute given the knowledge of another attribute.*

Example 1 Consider the employee database in Table 1, by Definition 2, we can compute $I(\texttt{age};\texttt{married}) = \sum_{v_{\texttt{age}} \in \{1,2,3,4,5\}} \sum_{v_{\texttt{married}} \in \{1,2\}} p(v_{\texttt{age}}, v_{\texttt{married}}) \log \frac{p(v_{\texttt{age}}, v_{\texttt{married}})}{p(v_{\texttt{age}})p(v_{\texttt{married}})} = 0.47$. This shows that the knowledge of `age` causes a reduction of 0.47 in the uncertainty of `married`. However, we have little idea how much a reduction of 0.47 is. Using the normalization, we can compute $\widetilde{I}(\texttt{age};\texttt{married}) = \frac{I(\texttt{age; married})}{MAX\{H(\texttt{age}), H(\texttt{married})\}} = \frac{I(\texttt{age; married})}{H(\texttt{age})} = \frac{0.47}{2.19} = 0.21$, which implies a reduction of at least 21% of the uncertainty of `age` and `married`.

Similarly, we can compute $I(\texttt{gender};\texttt{education}) = 0.40$ and $\widetilde{I}(\texttt{gender};\texttt{education}) = \frac{I(\texttt{gender; education})}{H(\texttt{education})} = \frac{0.40}{1.34} = 0.30$.

Note that $I(\texttt{age};\texttt{married}) > I(\texttt{gender};\texttt{education})$, but $\widetilde{I}(\texttt{age};\texttt{married}) < \widetilde{I}(\texttt{gender};\texttt{education})$. This means that the minimum percentage of reduction in the uncertainty of `gender` and `education` is higher than that of `age` and `married`, although the MI of the former is lower than that of the latter. The higher MI of `age` and `married` is mainly because the entropy of `age` is much higher than that of `education` (i.e., $H(\texttt{age}) = 2.19 > H(\texttt{education}) = 1.34$), which means a much larger absolute value of uncertainty to be reduced rather than the relative amount. This shows the advantage of NMI over MI. □

4. QUANTITATIVE CORRELATED PATTERNS

In this section, we first generalize the concept of all-confidence for a quantitative pattern and then propose the notion of quantitative correlated patterns.

There have been a number of measures [3, 12, 11] proposed for correlations. In recent years, *all-confidence* [12, 11] has emerged as a commonly adopted correlation measure and has been shown in many studies [12, 11, 18, 10, 9] that it reflects the true correlative relationship among attributes more accurately than do other measures. The all-confidence of a boolean pattern is defined as *the minimum confidence of all the association rules that can be derived from the pattern.* We generalize all-confidence for a quantitative pattern as follows.

Definition 4 (All-Confidence of a Quantitative Pattern) The *all-confidence* of a quantitative pattern X, denoted as $\mathit{allconf}(X)$, is defined as

$$\mathit{allconf}(X) = \frac{\mathit{supp}(X)}{MAX\{\mathit{supp}(x[l_x, u_x]) \mid x[l_x, u_x] \in X\}}.$$

A pattern is said to be interesting if its all-confidence is no less than a given *minimum all-confidence threshold* ς. According to this definition, any association rule derived from the pattern has confidence no less than ς, which also indicates a high correlation among all the items in the pattern (note that a high-confidence association rule only indicates an implication from the set of items at the left side of the rule to that at the other side).

All-confidence has the downward closure property [12], which means that if a pattern has all-confidence no less than ς, so do all its sub-patterns. This property also holds for the all-confidence of quantitative patterns since the sub-pattern in quantitative databases is defined in the same way as that in boolean databases.

Example 2 Given the employee database in Table 1, we consider the pattern $X = \texttt{gender}[1,1]\texttt{education}[1,1]$ and compute $\mathit{allconf}(X) = \frac{\mathit{supp}(\texttt{gender}[1,1]\texttt{education}[1,1])}{MAX\{\mathit{supp}(\texttt{gender}[1,1]),\mathit{supp}(\texttt{education}[1,1])\}}$ $= \frac{0.19+0.11+0.09+0.09}{MAX\{0.25+0.19+0.11+0.09+0.09+0.09+0.08,\ 0.19+0.11+0.09+0.09\}}$ $= 0.53$. Similarly, we can compute the all-confidence of the pattern $Y = \texttt{gender}[1,1]\texttt{married}[1,1]$ to be $\mathit{allconf}(Y) = 0.9$, which indicates a higher correlation among its items than that among the items of X. □

Although all-confidence is a good measure of correlation among boolean attributes, it is inadequate for reflecting the correlation among quantitative attributes. This is because all-confidence is a measure applied at a fine granularity to the intervals of attributes. However, quantitative attributes often consist of a large number of intervals, we may obtain patterns that have high all-confidence simply as a result of co-occurrence (see an example in Example 3). In this case, all-confidence cannot serve as a true measure of the correlation among the attributes in the pattern.

Realizing that the definition of a correlated pattern [3] is *a set of attributes that are dependent on each other* and that MI is a well-established concept in information theory [6] to *capture the dependency among attributes*, we incorporate the concept of MI into the definition of a QCP. In this way, we first ensure that every attribute in a QCP is strongly dependent on each other in the sense that every attribute carries a great amount of information about every other attribute in the pattern. Then, we further use all-confidence to guarantee that the intervals of the attributes are also highly correlated.

Definition 5 (Quantitative Correlated Pattern) Given a *minimum information threshold* μ $(0 \leq \mu \leq 1)$ and a *minimum all-confidence threshold* ς $(0 \leq \varsigma \leq 1)$, a pattern X is called a *Quantitative Correlated Pattern* (*QCP*) if and only if the following two conditions are satisfied:

1. $\forall x, y \in \mathit{attr}(X)$, $\widetilde{I}(x; y) \geq \mu$;
2. $\mathit{allconf}(X) \geq \varsigma$.

NMI has several properties that make it a natural measure of correlation. First, NMI is a formal concept for measuring dependency between attributes. Second, NMI gives an intuitive meaning for quantifying the degree of dependency: NMI has a value of 0 to indicate independence and its value increases, within the unit range, with the increase in dependency. Third, we can define a threshold μ for NMI to indicate the required minimum percentage of reduction in the uncertainty of an attribute given the knowledge of another attribute.

Example 3 Given the employee database in Table 1, let $\mu = 0.2$ and $\varsigma = 0.5$. The pattern $X = \texttt{gender}[1,1]\texttt{education}[1,1]$ is a QCP, since $\widetilde{I}(\texttt{gender}, \texttt{education}) = 0.30 \geq \mu$ as shown in Example 1, and $\mathit{allconf}(X) = 0.53 \geq \varsigma$ as shown in Example 2. However, the pattern $Y = \texttt{gender}[1,1]\texttt{married}[1,1]$ is not a QCP because $\widetilde{I}(\texttt{gender}, \texttt{married}) = 0 < \mu$, although $\mathit{allconf}(Y) = 0.9 \geq \varsigma$. The truth is that the attributes `gender` and `married` are independent of each other, which can be easily verified by $p(v_{\texttt{gender}}, v_{\texttt{married}}) = (p(v_{\texttt{gender}}) \cdot p(v_{\texttt{married}}))$ for every possible $v_{\texttt{gender}}$ and $v_{\texttt{married}}$. The reason for the high all-confidence of Y is simply because both $p(\texttt{gender}[1,1])$ and $p(\texttt{married}[1,1])$ are very high (both of them are 0.9), which results in a high co-occurrence of the two items $\texttt{gender}[1,1]$ and $\texttt{married}[1,1]$. Obviously, patterns such as Y are of little significance because they do not reveal the true correlations between the items in the patterns. This explains the necessity of the concept of NMI in the definition of QCPs. □

Problem Description Given a quantitative database $\mathcal{D}$, a minimum information threshold μ and a minimum all-confidence threshold ς, the mining problem we are going to solve in this paper is to find all QCPs from $\mathcal{D}$.

5. A SUPERVISED INTERVAL COMBINING METHOD

Before we mine the quantitative databases, we first discretize the databases, using a discretization method such as equi-depth and equi-width, in order to deal with the continuous values and the large domain sizes. We discretize each quantitative attribute into a set of *base intervals*, each of which is assigned a label. The base intervals are considered as indivisible units during the mining process. Consecutive base intervals may be combined into larger intervals to gain sufficient support value, while a combined interval itself can have a more significant meaning than its composite base intervals. However, it is critical to control the interval combining process to avoid a combined interval becoming too trivial. For example, `age`$[0,2]$ refers to infants and is more representative than `age`$[0,0]$, `age`$[1,1]$ or `age`$[2,2]$; however, `age`$[0,100]$ is simply trivial.

The traditional method of controlling the size of a combined interval using a maximum support threshold [15] is inapplicable in our problem of mining correlations. This is because QCPs can be both rare patterns (of low support) and popular patterns (of high support) and thus have a wide range of support value. Other more sophisticated interval combining methods such as [16] have also been proposed but are primarily concerned with mining quantitative association rules.

In mining QCPs, it would be advantageous to consider the dependency between the attributes when combining their intervals, because the intervals of an attribute can be combined in very different ways with respect to different attributes as to reflect specific meanings. For example, combining the intervals of the attribute `age` with respect to `married` can obtain totally different combined intervals compared to that with respect to `gender`, which is further elaborated in Example 4.

We find that MI can be used to take into account the dependency between attributes and thus to guide the interval combining process to produce meaningful combined intervals. Since the interval combining is performed locally between a pair of attributes, we use MI, instead of NMI which is a global measure for all attributes.

We model the interval combining problem as a supervised optimization problem with MI as the objective function, which is described as follows.

Given two attributes x and y, where x is quantitative and y can either be categorical or quantitative, we want to obtain the optimal combined intervals of x with respect to y. The objective function, ϕ, of the optimization problem is defined as follows:

$$\begin{aligned}\phi(x,y) &= I'(x;y) - I(x;y) \\ &= (H'(x) + H(y) - H'(x,y)) \qquad \text{By Property 6} \\ &\quad -(H(x) + H(y) - H(x,y)) \\ &= (H'(x) - H(x)) + (H(x,y) - H'(x,y)), \qquad (1)\end{aligned}$$

where $I'(x;y)$, $H'(x)$ and $H'(x,y)$ are the respective values of MI and entropy after combining the intervals of x. Note that $H(y)$ remains unchanged, because the intervals of y are not combined.

Since both $H(x)$ and $H(x,y)$ always decrease when the intervals of x are combined, ϕ can be either positive or negative depending on the rate of decrease of $H(x)$ and $H(x,y)$. Thus, the optimization problem is to maximize the function ϕ, that is, to either maximize the gain in MI (if $\phi > 0$) or minimize the loss in MI (if $\phi < 0$).

We now design an algorithm to solve this optimization problem. If x has n base intervals, to find an optimal solution will require $\mathcal{O}(2^n)$ computations of MI values, where each MI value is computed from a possible set of combined intervals. Obviously, an exhaustive algorithm is unrealistic. We propose an efficient algorithm which greedily combines two consecutive intervals of x at each time. The idea of the greedy algorithm is described as follows.

At each time, we consider combining two consecutive intervals, i_{x_1} and i_{x_2}, of x, where i_{x_1} and i_{x_2} can be either a base interval or a combined interval. Let $\phi_{[i_{x_1},i_{x_2}]}(x,y)$ denote the value of $\phi(x,y)$ when i_{x_1} and i_{x_2} are combined with respect to y.

Our algorithm, *GreedyCombine*, is shown in Procedure 1. The idea (Steps 13-19) is to pick up at each time the maximum $\phi_{[i_{x_j},i_{x_{j+1}}]}(x,y)$ among all pairs of consecutive intervals, i_{x_j} and $i_{x_{j+1}}$, and combine corresponding i_{x_j} and $i_{x_{j+1}}$ into $i_{x_{j'}}$. Then, $\phi_{[i_{x_{j-1}},i_{x_j}]}(x,y)$ and $\phi_{[i_{x_{j+1}},i_{x_{j+2}}]}(x,y)$ are replaced by $\phi_{[i_{x_{j-1}},i_{x_{j'}}]}(x,y)$ and $\phi_{[i_{x_{j'}},i_{x_{j+2}}]}(x,y)$.

Algorithm 1 *CombineInterval*()

```
1. for each quantitative attribute $x$ do
2.   for each attribute $y \neq x$, and the pair $x$ and $y$ has not
     been considered do
3.     $GreedyCombine(x, y, -\infty, -\infty, 0)$;
4.     Output the intervals of $x$ and $y$ after interval combining;
```

Procedure 1 *GreedyCombine*$(x, y, \phi^x_{min}, \phi^y_{min}, flag)$

```
1.  for each pair of consecutive intervals $i_{x_j}$ and $i_{x_{j+1}}$ of $x$ do
2.    if ($\phi_{[i_{x_j},i_{x_{j+1}}]}(x,y) \geq \phi^x_{min}$)
3.      Insert $\phi_{[i_{x_j},i_{x_{j+1}}]}(x,y)$ into a heap, $Q$;
4.  if ($Q$ is empty)            \\no intervals of $x$ can be combined
5.    if ($flag = 1$)            \\no intervals of $y$ can be combined
6.      Terminate;
7.    else                       \\$flag = 0$
8.      $flag \leftarrow 1$;     \\set flag for the next iteration
9.      Goto Step 21;
10. else                         \\$Q$ is not empty
11. $\phi^x_{min} \leftarrow MEAN\{\phi_{[i_{x_j},i_{x_{j+1}}]}(x,y) \in Q\}$;
12. $flag \leftarrow 0$;         \\reset flag for the next iteration
13. Extract maximum $\phi_{[i_{x_j},i_{x_{j+1}}]}(x,y)$ from $Q$;
14. if ($\phi_{[i_{x_j},i_{x_{j+1}}]}(x,y) \geq \phi^x_{min}$)
15.   Combine $i_{x_j}$ and $i_{x_{j+1}}$ into $i_{x_{j'}}$;
16.   $\phi_{[i_{x_{j-1}},i_{x_j}]}(x,y) \leftarrow \phi_{[i_{x_{j-1}},i_{x_{j'}}]}(x,y)$;
17.   $\phi_{[i_{x_{j+1}},i_{x_{j+2}}]}(x,y) \leftarrow \phi_{[i_{x_{j'}},i_{x_{j+2}}]}(x,y)$;
18.   Update $Q$;
19.   Goto Step 13;
20. else                         \\no more intervals of $x$ can be combined
21.   if ($y$ is quantitative)
22.     $GreedyCombine(y, x, \phi^y_{min}, \phi^x_{min}, flag)$;
23.   else
24.     Terminate;
```

We can efficiently retrieve the maximum $\phi_{[i_{x_j},i_{x_{j+1}}]}(x,y)$ by implementing a priority queue using a heap Q (Step 3), while $\phi_{[i_{x_{j-1}},i_{x_j}]}(x,y)$ and $\phi_{[i_{x_{j+1}},i_{x_{j+2}}]}(x,y)$ can be accessed by putting two pointers in $\phi_{[i_{x_j},i_{x_{j+1}}]}(x,y)$. The update of their positions in the heap takes only $\mathcal{O}(\log n)$ *heapify* operations. In the worst case when all intervals of x are to be combined into a single interval, the entire combining process takes $\mathcal{O}(n \log n)$ heapify operations ($\mathcal{O}(1)$ each) and $\mathcal{O}(n)$ computations of $\phi_{[i_{x_j},i_{x_{j+1}}]}(x,y)$ ($\mathcal{O}(l)$ each), where n and l are the number of base intervals of x and y, respectively.

To avoid a combined interval becoming too trivial, we set a terminating condition, ϕ^x_{min}, as follows. We first set ϕ^x_{min} to be the mean of all $\phi_{[i_{x_j},i_{x_{j+1}}]}(x,y)$ in the heap (Step 11) with extremely small values removed. This initial value of ϕ^x_{min} is chosen in order to allow most pairs of consecutive intervals, that have relatively high ϕ, to have a chance to be combined before we start combining. Thus, ϕ^x_{min} serves as the minimum gain in MI (if $\phi^x_{min} > 0$) or the maximum loss in MI (if $\phi^x_{min} < 0$) that we require. When intervals are combined, the heap is updated (Step 18) and some $\phi_{[i_{x_j}, i_{x_{j+1}}]}(x, y)$ may become less than ϕ^x_{min}. As a result, the corresponding i_{x_j} and $i_{x_{j+1}}$ will not be combined (Step 20).

If both attributes x and y are quantitative, we combine the intervals of x and y in turn recursively (Steps 21-22). The *GreedyCombine* terminates when the intervals of both attributes cannot be combined any more with respect to their respective ϕ_{min} (Steps 6 and 24). We keep a boolean *flag* as one of the input parameters of *GreedyCombine* to indicate whether the intervals of the other attribute are combined or not in the last iteration.

The main algorithm, *CombineInterval*, to combine the intervals for pairs of attributes, is shown in Algorithm 1. For each pair of attributes, which contains at least one quantitative attribute, *CombineInterval* invokes *GreedyCombine* by initializing ϕ^x_{min} and ϕ^y_{min} to be $-\infty$ and *flag* to be 0.

Example 4 Consider the employee database in Table 1, where each label of quantitative attributes corresponds to one base interval. Using *GreedyCombine*, the combined intervals of `age` with respect to `married` are $[1, 1]$ and $[2, 5]$. This is reasonable because for the transactions with `married` = 2, they all have a value of 1 for `age`. While for other transactions with `married` = 1, their values of `age` fall within the interval $[2, 5]$. This reflects the case in real life that most of the people over a certain age, say 35, are married.

However, if we compute the combined intervals of `age` with respect to `gender`, the results are $[1, 2]$, $[3, 4]$ and $[5, 5]$, which are totally different from those of `age` with respect to `married`. Fewer base intervals of `age` are combined with respect to `gender`, because for the transactions with the same value of `gender`, their values of `age` scatter over all its possible values. This shows the case that there are young, middle-aged and old employees of both men and women. □

6. MINING QUANTITATIVE CORRELATED PATTERNS

In this section, we present our algorithm of mining QCPs. Our algorithm utilizes NMI and all-confidence to perform a two-level pruning, which significantly reduces the search space of the mining problem. We first describe the pruning at each level and then present the overall algorithm.

6.1 Attribute-Level Pruning

The first condition of Definition 5 requires that, to generate a pattern in the mining process, the NMI of every pair of attributes in the pattern must be at least μ. This condition enables us to perform pruning at the attribute level of the mining problem. We show how the pruning is performed by introducing the NMI graph as follows.

Definition 6 (Normalized Mutual Information Graph) A *Normalized Mutual Information graph* (*NMI graph*) is an undirected graph, $G = (V, E)$, where $V = \mathcal{I}$ is the set of nodes and $E = \{(x_i, x_j) \mid x_i \neq x_j \text{ and } \widetilde{I}(x_i, x_j) \geq \mu\}$ is the set of edges.

Lemma 1 (Necessary Condition) If X is a QCP, then *attr*(X) forms a clique in G.

Proof. It follows directly from Definitions 5 and 6. □

The necessity that the attribute set of a QCP must form a clique in the NMI graph reveals the strong inter-dependence between all attributes in a QCP.

Lemma 1 implies that we can generate the attribute sets of all QCPs by enumerating the cliques in the NMI graph. Since the mining approach without pruning at the attribute level can be modelled as a complete graph, that is, an NMI graph with $\mu = 0$, the search space is greatly reduced from enumerating all cliques in the complete graph to enumerating all cliques in a much sparser NMI graph. The significance of this pruning at the attribute level is fully disclosed if we realize that an edge in the NMI graph can generate an enormous number of patterns, which is equal to the size of the cartesian product of the set of intervals of two incident nodes (i.e., attributes) of the edge. We illustrate this concept in further detail in Section 6.2.

The complexity of enumerating all cliques in a graph is exponential. However, we show that the clique enumeration can be seamlessly incorporated into the mining process. Our mining algorithm adopts a prefix tree structure, called the *attribute prefix tree*, denoted as T_{attr}, which is constructed as follows.

First, a root node is created at Level 0 of T_{attr}. Then at Level 1, we create a node for each attribute in $\mathcal{I}$ as a child of the root, where each child node is assigned a label as the label of the attribute and the order of the children follows that of the attributes in $\mathcal{I}$. T_{attr} is then constructed in a depth-first manner as follows. For each node u at Level k ($k \geq 1$) and for each right sibling v of u, if (u, v) is an edge in G, we create a child node for u with the same attribute label as that of v. Then, we continue the construction in the same way with u's children at Level $(k + 1)$.

Lemma 2 Let $\langle u_1, \ldots, u_k \rangle$ be a path from a node u_1 at Level 1 to a node u_k at Level k of T_{attr}. Then, $\{u_1, \ldots, u_k\}$ forms a clique in G.

Proof Sketch. By induction on the length of the path, k. □

The prefix tree is shown to be a very efficient data structure for mining both frequent and correlated patterns, while Lemma 2 shows that the clique enumeration comes almost free with the construction of T_{attr}. The only extra processing incurred is a trivial test of whether (u, v) is an edge in G. Moreover, the clique enumeration can be terminated earlier by the all-confidence pruning described in the following subsection.

Lastly, we provide an easy and objective way of setting the minimum information threshold μ as in Equation (2), which is the sum of the mean and the standard deviation of all distinct NMI values, so that G retains edges that reveal high mutual dependency between the two incident nodes. We also remark that, similar to the choice of thresholds for other measures such as the minimum support threshold in the frequent pattern mining problem, the choice of μ can also be determined by domain experts to indicate how correlated they require the attributes in a pattern to be.

$$\mu = MEAN\{\widetilde{I}(x; y) \mid x \neq y\} + STD\{\widetilde{I}(x; y) \mid x \neq y\} \quad (2)$$

Example 5 Given the employee database in Table 1, based on the combined intervals of quantitative attributes, we compute the NMI graph G as shown in Figure 1, where $\mu = 0.26$ as given by Equation (2). There are only four edges in G, each of which is identified as a strong dependency between two attributes. Other edges do not exist in G because they

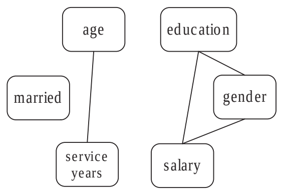


Figure 1: An NMI Graph G

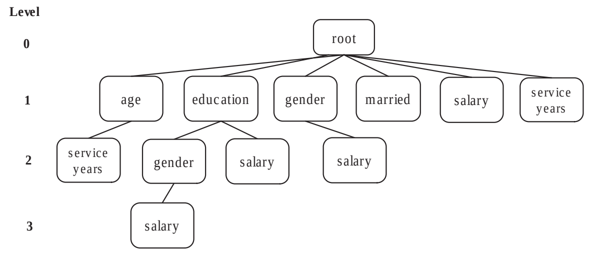


Figure 2: An Attribute Prefix Tree T_{attr}

cannot constitute any QCP. This ensures that uncorrelated patterns, such as `gender`$[1,1]$`married`$[1,1]$ in Example 3, will not be generated, because there is no edge between `gender` and `married` in G.

To find the cliques in G, we construct an attribute prefix tree T_{attr} as shown in Figure 2. It can be easily verified that each k-path in T_{attr} represents a k-clique in G. $\Box$

6.2 Interval-Level Pruning

Although NMI can effectively eliminate the generation of patterns from uncorrelated attributes, patterns with low all-confidence may still be generated from correlated attributes. This is because a node in the attribute prefix tree T_{attr} actually represents a set of patterns that have the same attribute set but different interval set. Thus, we also need pruning at the interval level. For this purpose, we employ the downward closure property [12] of all-confidence to prune a pattern X and all its super-patterns if $allconf(X) < \varsigma$.

Since the intervals of an attribute are combined in a supervised way, the same attribute may have different set of combined intervals with respect to different attributes. When we join two k-patterns to produce a $(k+1)$-pattern, the intervals of the prefixing $(k-1)$ attributes in the two k-patterns may overlap. In this case, an easy way is to compute the intersection of the prefixing $(k-1)$ intervals of the two k-patterns to give the intervals for the $(k+1)$-pattern. For example, given `age`$[30,40]$`married`$[1,1]$ and `age`$[25,35]$`salary`$[2000,3000]$, we intersect the intervals of `age` to obtain the new pattern `age`$[30,35]$`married`$[1,1]$`salary` $[2000,3000]$.

However, the power of pruning by all-confidence comes from its downward closure property. Producing a $(k+1)$-pattern by intersecting the intervals of k-patterns violates the downward closure property of all-confidence. This is because shrinking the intervals in the $(k+1)$-pattern may cause a great decrease in the support value of a single item so that the all-confidence of the $(k+1)$-pattern may become larger than that of its composite k-patterns. However, this problem can be addressed if we enumerate all sub-intervals of a (combined) interval before we start to generate a pattern.

We first define the *sub-interval* of an interval. Given an interval $[l,u]$, a *sub-interval* of $[l,u]$ is an interval $[l',u']$, where $l \leq l' \leq u' \leq u$. We use $[l',u'] \sqsubseteq [l,u]$ to denote $[l',u']$ is a sub-interval of $[l,u]$.

Recall that a node at Level k of T_{attr} represents a k-attribute set. We start from Level 2 of T_{attr} to generate 2-patterns. Let $\{x,y\}$ be the attribute set represented by a node at Level 2, and S_x and S_y be the set of combined intervals of x and y. Similar to mining quantitative frequent patterns [15], we need to consider all pairs of sub-intervals of x and y as each of them represents a pattern. For each interval set $\{i'_x, i'_y\}$, where $i'_x \sqsubseteq i_x$, $i'_y \sqsubseteq i_y$, $i_x \in S_x$ and $i_y \in S_y$, we generate a QCP $X = x[i'_x]y[i'_y]$ if $allconf(X) \geq \varsigma$.

The above computation is performed on the cartesian product of two sets of sub-intervals of x and y. The size of the cartesian product can be very big since an interval $i_x = [l, l+n]$ has $\frac{n(n+1)}{2}$ sub-intervals. Fortunately, our supervised interval combining method effectively clusters the base intervals of an attribute into small groups, which drastically reduces the size of the cartesian product.

Since the intersection of two overlapping intervals is just a common sub-interval of the two intervals, we ensure that all QCPs will be generated by enumerating all pairs of sub-intervals. Moreover, since all the possible sub-interval combinations are considered in 2-patterns, which are the basis for generating k-patterns $(k > 2)$, the downward closure property of all-confidence holds as usual and can be applied to perform the pruning. The sub-intervals are then considered as indivisible intervals during the mining process and not intersected.

For a set of k-patterns generated at a node at Level k $(k \geq 2)$ of T_{attr}, they often share a large number of common sub-intervals in their prefixing $(k-1)$-interval sets. Thus, we also use a prefix tree $T^u_{interval}$, called the *interval prefix tree*, to keep the interval sets of all the patterns generated by a node u in T_{attr}. The interval prefix tree not only saves memory for storing the duplicate sub-intervals, but also significantly speeds up the join of two k-patterns to produce a $(k+1)$-pattern.

6.3 QCoMine Algorithm

We now present our main algorithm, *QCoMine*, as shown in Algorithm 2. We first combine the base intervals of each quantitative attribute with respect to another attribute. Then we construct the NMI graph G and use G to guide the construction of the attribute prefix tree T_{attr} to perform pruning at the attribute level. Steps 5-13 of Algorithm 2 construct Level 2 of T_{attr} and produce all 2-QCPs. Steps 14-15 invoke *RecurMine*, as shown in Procedure 2, to generate all k-QCPs $(k > 2)$ recursively in a depth-first manner. Note that at Step 6 of *RecurMine* when two k-patterns are joined, all the prefixing $(k-1)$ intervals should be the same in the two patterns, which means that no interval intersection is performed; in addition, the last intervals of two k-patterns should form the interval set of a corresponding 2-pattern, so as to ensure that the last interval is a sub-interval of one attribute with respect to the other.

To compute the all-confidence of a pattern, we adopt *diffset* [19] to obtain the support value of the pattern, while we use an extra field to keep the maximum support value of the items in the pattern. The use of *diffset*, together with the depth-first strategy, effectively controls memory consumed in the mining process as evidenced by our experiments.

Algorithm 2 *QCoMine*($\mathcal{D}$, μ, ς)

```
1.  CombineInterval();
2.  Construct the NMI graph $G$;
3.  Create the root node, root, of $T_{attr}$;
4.  Create a node for each attribute in $\mathcal{I}$ as a child of root;
5.  for each child node $u$ of root do
6.    for each right sibling $v$ of $u$ do
7.      if $((u,v) \in G)$
8.        Create $w$ as a child of $u$ and assign to $w$ an attribute
           label the same as that of $v$ in $T_{attr}$;
9.        Let $\{x, y\}$ be the attribute set represented by $w$;
10.       for each sub-interval pair, $i_x$ and $i_y$, of $x$ and $y$ do
11.         if $(allconf(X = x[i_x]y[i_y]) \geq \varsigma)$
12.           Output $X$ as a QCP;
13.           Insert $X$'s interval set $\{i_x, i_y\}$ into $T^w_{interval}$;
14.   for each child node $w$ of $u$ do
15.     RecurMine($w, T_{attr}, G, 2$);
```

Procedure 2 *RecurMine*(u, T_{attr}, G, k)

```
1.  for each right sibling $v$ of $u$ do
2.    if $((u,v) \in G)$
3.      Create $w$ as a child of $u$ and assign to $w$ an attribute
         label the same as that of $v$ in $T_{attr}$;
4.      Let $\{x_1, \ldots, x_{k+1}\}$ be the attribute set represented by $w$;
5.      Let $\{i_{u_1}, \ldots, i_{u_{k-1}}, i_{u_k}\}$ and $\{i_{v_1}, \ldots, i_{v_{k-1}}, i_{v_k}\}$ be any
         two interval sets in $T^u_{interval}$ and $T^v_{interval}$;
6.      if $(i_{u_j} = i_{v_j}$ for $1 \leq j \leq k-1$, and $\{i_{u_k}, i_{v_k}\}$ is
          an interval set of the attribute set $\{x_k, x_{k+1}\})$
7.        Let $X = x_1[i_{u_1}] \ldots x_{k-1}[i_{u_{k-1}}]x_k[i_{u_k}]x_{k+1}[i_{v_k}]$;
8.        if $(allconf(X) \geq \varsigma)$
9.          Output $X$ as a QCP;
10.         Insert $\{i_{u_1}, \ldots, i_{u_{k-1}}, i_{u_k}, i_{v_k}\}$ into $T^w_{interval}$;
11. Delete $T^u_{interval}$;
12. for each child node $w$ of $u$ do
13.   RecurMine($w, T_{attr}, G, k+1$);
```

Example 6 (Example 5 continued) Let $\mu = 0.26$ and $\varsigma = 0.6$. The T_{attr} constructed by *QCoMine* is shown in Figure 2. The node gender at Level 2 of T_{attr} represents the 2-attribute set {education, gender}. Both education and gender are categorical, thus all the sub-interval pairs of this attribute set are the six combinations of three values of education and two values of gender. Among the six corresponding 2-patterns, only education$[3,3]$gender$[2,2]$ has all-confidence of 0.9, which is greater than ς.

The node salary at Level 2 of T_{attr}, which is the child of the node education, represents the 2-attribute set {education, salary}. The combined intervals of salary with respect to education are $[1,1]$, $[2,3]$, $[4,4]$, which have five sub-intervals: $[1,1]$, $[2,2]$, $[2,3]$, $[3,3]$, $[4,4]$. Thus, there are fifteen sub-interval pairs formed for education and salary, among which only one corresponding pattern education$[3,3]$salary$[4,4]$ satisfies the all-confidence.

The node salary at Level 3 is generated by the *RecurMine* procedure, which joins the two 2-patterns education$[3,3]$ gender$[2,2]$ and education$[3,3]$salary$[4,4]$ to produce a 3-pattern education$[3,3]$gender$[2,2]$salary$[4,4]$, which has all-confidence of 0.9. □

7. PERFORMANCE EVALUATION

We evaluate the performance of our approach of mining correlations from quantitative databases on real datasets. All experiments were run on a linux machine with an AMD Opteron 844 (1.8GHz) CPU and 8 GB RAM.

We use two real datasets from the commonly used UCI machine learning repository [8]. Table 3 lists the name, the number of transactions, the number of attributes, and the maximum number of base intervals after the discretization, of each dataset. The number of quantitative attributes of each dataset is given in the brackets. The detailed information of these datasets can be found in [8].

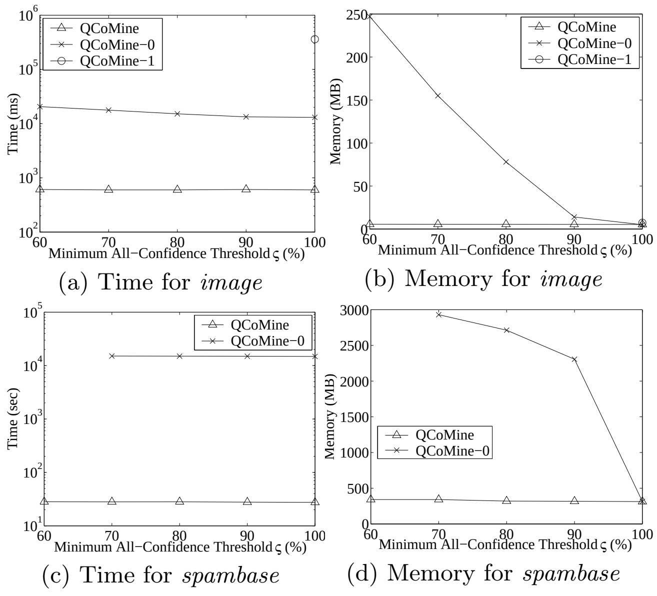
Table 3: Dataset Description

Dataset	Transactions	Attributes (Quantitative)	Maximum Base Intervals
image	2,310	20(19)	96
spambase	4,601	58(57)	761

7.1 Performance of QCoMine

The efficiency of our algorithm *QCoMine* and the quality of our QCPs are based on three major components that constitute *QCoMine*: the supervised interval combining method, the attribute-level pruning by NMI and the interval-level pruning by all-confidence. Since there is no existing work on mining correlations from quantitative databases, we mainly assess the effect of these three components on the performance of our approach.

We make three variants of our algorithm: (a) QCoMine, which applies the interval combining method and sets μ as described by Equation (2); (b) QCoMine-0, which applies the interval combining method and sets $\mu = 0$; and (c) QCoMine-1, which does not apply the interval combining method and sets μ as described by Equation (2). We test all-confidence from $\varsigma = 60\%$ to $\varsigma = 100\%$.



(a) Time for *image* (b) Memory for *image*
(c) Time for *spambase* (d) Memory for *spambase*

Figure 3: Performance of QCoMine

Effect of Supervised Interval Combining. When the interval combining method is not applied, we are only able to obtain the result on the dataset *image* at $\varsigma = 100\%$ as shown in Figures 3(a-b), while QCoMine-1 runs out of memory on all other cases. QCoMine-1 is inefficient because when we allow the interval of an item to become too trivial, patterns will easily gain all-confidence greater than ς by co-occurrence in the database. The number of patterns obtained by QCoMine-1 from *image* at $\varsigma = 100\%$ is 13.4 times more than that obtained by QCoMine and the difference increases rapidly for smaller ς (700M patterns are returned at $\varsigma = 90\%$ before QCoMine-1 runs out of memory).

We define the *span* of an interval, $[l, u]$, of an attribute as $\frac{u-l}{n}$, where n is the number of base intervals of the attribute. For example, if age has 100 base intervals, the interval span of $[20, 80]$ is 60%. We find that, for the patterns returned by QCoMine-1 but not by QCoMine, 35% of them consist of intervals that have a span of 90% (i.e., almost the entire domain), while most of the rest consist of at least one interval with a span over 30%.

The results show that our supervised interval combining method is effective in defining more meaningful intervals and thus avoids an overwhelming number of trivial patterns being mined, which is essential in the control of memory and CPU usage.

Effect of Normalized Mutual Information. The performance improvement by utilizing NMI as a pruning tool is clearly revealed by the performance difference between QCoMine and QCoMine-0. Figures 3(a-d) show that, compared with QCoMine-0, the running time of QCoMine is reduced by over one order of magnitude for *image* and almost three orders of magnitude for *spambase*, while QCoMine consumes memory up to 44 times less for *image* and 10 times less for *spambase* than does QCoMine-0.

The number of patterns obtained by QCoMine is on average 65 times less for *image* and 88 times less for *spambase* than that obtained by QCoMine-0. The extra patterns returned by QCoMine-0 are shown to consist of attributes with large interval spans. Recall that QCoMine-0 also applies our interval combining method. However, the result does not mean that the interval combining method is not effective. We investigate the attributes in the datasets and find that, if an attribute x has no or little correlation with another attribute y, our interval combining method may return some rather trivial combined intervals for x with respect to y. Such uncorrelated patterns are successfully pruned by the use of NMI in QCoMine and thus not returned.

Therefore, the results demonstrate the effectiveness of NMI both as a measure for correlation and as a tool for pruning unpromising search space.

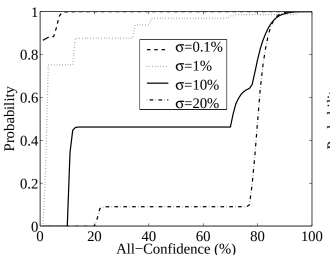
Effect of All-Confidence. Figures 3(a) and 3(c) show that the running time of both QCoMine and QCoMine-0 increases only slightly for smaller ς. This is because the majority of the time is spent on computing the 2-patterns. No matter what the value of ς is, we need to test every 2-pattern to determine whether it is a QCP, before we can employ the downward property of all-confidence to prune all super-patterns of an uncorrelated pattern. Therefore, the slight difference in running time for different values of ς in fact reflects the pruning power of all-confidence, since our algorithm only spends a small portion of the time to generate the larger patterns when the pruning starts to work.

The number of patterns returned by QCoMine grows steadily by about 2 times for each decrease in ς from 100% to 60%. A similar trend is also observed for QCoMine-0, except that when ς decreases from 100% to 90%, there is a rapid increase in the number of patterns that consist of attributes with large interval spans.

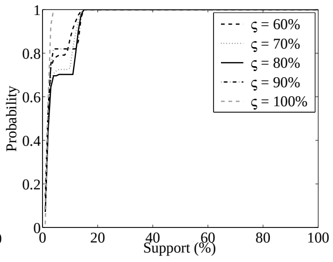
7.2 Quantitative Correlated Patterns v.s. Quantitative Frequent Patterns

In this section, we demonstrate the high complexity of mining quantitative frequent patterns as to further justify the effectiveness of our approach of mining correlations. We implement the algorithm proposed by Srikant and Agrawal [15] using the same prefix tree structure and the *diffset* [19] as used in QCoMine. We denote this algorithm as MFP in this experiment.

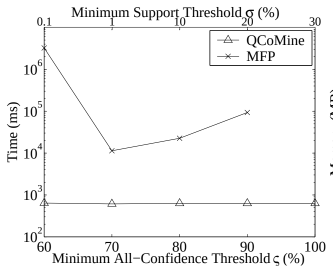
We test five settings of *minimum support threshold* $\sigma = 0.1\%, 1\%, 10\%, 20\%, 30\%$, for MFP. Since MFP uses a *maximum support threshold*, σ_m, to control the span of a combined interval, we set $\sigma_m = 1.3\sigma$, which means the support of a combined interval is at most 1.3σ.



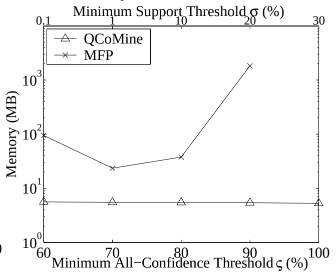
(a) All-Confidence Distribution for MFP



(b) Support Distribution for QCoMine



(c) Time



(d) Memory

Figure 4: Quantitative Correlated Patterns v.s. Quantitative Frequent Patterns for *image*

Figure 4(a) presents the cumulative probability distribution of all-confidence over the patterns obtained by MFP for *image*. When σ is small ($\leq 1\%$), over 80% of the patterns have very low all-confidence of less than 10%. When $\sigma = 10\%$, there are still half of the patterns having all-confidence of only 10%. Although most of the patterns have all-confidence greater than 80% when $\sigma = 20\%$, these patterns are mostly composed of attributes with trivial intervals, which are unlikely to be considered as useful knowledge. On the contrary, the support distribution of the patterns obtained by QCoMine, as presented in Figure 4(b), shows that most of the QCPs are rare (as over 70% of them have support less than 2%) and significant (as the items in these patterns are highly correlated). Mining such patterns using MFP requires a small σ, while MFP with a small σ may return a large number of uncorrelated patterns.

We also show the running time and memory consumption of MFP at each σ (as indicated by the upper x-axis) and QCoMine at each ς (as indicated by the lower x-axis) in Figures 4(c-d). Although they are incomparable, the figures do reflect that mining QCPs is much more stable in the use of resources than mining quantitative frequent patterns.

We do not present the results of MFP for *spambase* because MFP runs out of memory for all values of σ, even when we set σ_m almost the same as σ. At the point that the memory is exhausted, MFP already returns millions of patterns, which occupies over 20GB of disk space (for each σ). The massive number of patterns generated not only results in high memory consumption, but also reveals the difficulty in the use of the patterns for further analysis. On the contrary, QCoMine obtains impressive results for *spambase* as shown in Figures 3(c-d), which further reveals the effectiveness of mining QCPs over mining quantitative frequent patterns.

We also note that the high memory consumption of MFP is not due to our implementation, since MFP and QCoMine adopt the same depth-first strategy using the same data structure. In fact, the efficiency of QCoMine is primarily due to the supervised interval combining method and the pruning by NMI and all-confidence, as implied by the poor performance of QCoMine-0 and QCoMine-1 in Section 7.1.

8. RELATED WORK

Existing research on mining quantitative databases have mainly focused on mining quantitative association rules. This is first studied by Piatetshy-Shapiro [13] with both sides of the rule restricted to a single attribute. Srikant and Agrawal [15] generalize the work by allowing multiple attributes on both sides of the rule. Then, some variants of mining association rules have also been proposed, such as mining optimized association rules [7, 4, 14] by finding the optimal values of certain given attributes, and mining association rules with its consequent as a statistical measure [2, 17, 20] (e.g., mean, min, max) of a quantitative attribute.

Wang et al. [16] propose an interestingness-based criterion to merge intervals. Their merging criterion is based on association rules, which means that the candidate rules should be generated beforehand and the interval combining is then performed on the rules instead of the attributes. Our objective function, in contrast, is based on the attribute sets, which further guide the generation of QCPs.

In mining correlations from boolean databases, Brin et al. [3] introduce the correlation measures, χ^2 and *interest*. Cohen et al. [5] mine highly correlated 2-patterns measured by a symmetric similarity between two boolean attributes. Ma and Hellerstein [11] propose an m-pattern, of which any two subsets are mutually dependent measured by the conditional probability. Omiecinski [12] proposes two interesting measures, *all-confidence* and *bond*, both of which have the downward closure property. Xiong et al. [18] develop a measure called *h-confidence*, which is mathematically equivalent to all-confidence but defined from a different perspective to capture the degree of affinity in a pattern and to eliminate the cross-support patterns. Later in [10] and [9], all-confidence is shown to be a better measure for correlations than χ^2 and *interest*.

9. CONCLUSIONS

To our knowledge, our paper is the first study on mining correlations from quantitative databases. We propose a new notion of QCPs, which achieves two levels of quality control on correlation based on NMI and all-confidence. We develop a supervised interval combining method to combine the intervals according to the dependency between the attributes. We devise an efficient algorithm, *QCoMine*, to mine QCPs by utilizing NMI and all-confidence to perform a two-level pruning. Experimental results reveal that our interval combining method derives meaningful intervals and effectively eliminates the generation of trivial intervals, the number of which is always too large for the mining to be efficient. Our experiments also demonstrate that NMI is both an effective measure of correlation and a powerful tool for pruning unpromising search space arising from uncorrelated patterns, while all-confidence further ensures a stable performance for *QCoMine* as well as the quality of QCPs. We also show that *QCoMine* attains impressive speed and small memory consumption even when mining quantitative frequent patterns becomes too expensive, while the QCPs obtained are shown to be more useful than quantitative frequent patterns.

10. REFERENCES

[1] R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In *SIGMOD*, 1993.

[2] Y. Aumann and Y. Lindell. A statistical theory for quantitative association rules. *Journal of Intelligent Information Systems*, 20(3):255–283, 2003.

[3] S. Brin, R. Motwani, and C. Silverstein. Beyond market baskets: generalizing association rules to correlations. In *SIGMOD*, pages 265–276, 1997.

[4] S. Brin, R. Rastogi, and K. Shim. Mining optimized gain rules for numeric attributes. In *KDD*, pages 135–144, 1999.

[5] E. Cohen, M. Datar, S. Fujiwara, A. Gionis, P. Indyk, R. Motwani, J. D. Ullman, and C. Yang. Finding interesting associations without support pruning. *IEEE TKDE*, 13(1):64–78, 2001.

[6] T. M. Cover and J. A. Thomas. *Elements of Information Theory.* John Wiley & Sons, Inc., 1991.

[7] T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama. Data mining with optimized two-dimensional association rules. *ACM TODS*, 26(2):179–213, 2001.

[8] S. Hettich, C. Blake, and C. Merz. UCI repository of machine learning databases.

[9] W.-Y. Kim, Y.-K. Lee, and J. Han. Ccmine: Efficient mining of confidence-closed correlated patterns. In *PAKDD*, pages 569–579, 2004.

[10] Y.-K. Lee, W.-Y. Kim, Y. D. Cai, and J. Han. Comine: Efficient mining of correlated patterns. In *ICDM*, page 581, 2003.

[11] S. Ma and J. L. Hellerstein. Mining mutually dependent patterns. In *ICDM*, pages 409–416, 2001.

[12] E. R. Omiecinski. Alternative interest measures for mining associations in databases. *IEEE TKDE*, 15(1):57–69, 2003.

[13] G. Piatetsky-Shapiro. Discovery, analysis, and presentation of strong rules. In *Knowledge Discovery in Databases*, pages 229–248. 1991.

[14] R. Rastogi and K. Shim. Mining optimized association rules with categorical and numeric attributes. *IEEE TKDE*, 14(1):29–50, 2002.

[15] R. Srikant and R. Agrawal. Mining quantitative association rules in large relational tables. In *SIGMOD*, 1996.

[16] K. Wang, S. H. W. Tay, and B. Liu. Interestingness-based interval merger for numeric association rules. In *KDD*, pages 121–128, 1998.

[17] G. I. Webb. Discovering associations with numeric variables. In *KDD*, pages 383–388, 2001.

[18] H. Xiong, P.-N. Tan, and V. Kumar. Mining strong affinity association patterns in data sets with skewed support distribution. In *ICDM*, page 387, 2003.

[19] M. J. Zaki and K. Gouda. Fast vertical mining using diffsets. In *KDD*, pages 326–335, 2003.

[20] H. Zhang, B. Padmanabhan, and A. Tuzhilin. On the discovery of significant statistical quantitative rules. In *KDD*, pages 374–383, 2004.