Correlated Pattern Mining in Quantitative Databases

YIPING KE, JAMES CHENG, and WILFRED NG
The Hong Kong University of Science and Technology

We study mining correlations from quantitative databases and show that this is a more effective approach than mining associations to discover useful patterns. We propose the novel notion of Quantitative Correlated Pattern (QCP), which is founded on two formal concepts, mutual information and all-confidence. We first devise a normalization on mutual information and apply it to the problem of QCP mining to capture the dependency between the attributes. We further adopt all-confidence as a quality measure to ensure, at a finer granularity, the dependency between the attributes with specific quantitative intervals. We also propose an effective supervised method that combines 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 mine QCPs efficiently by utilizing normalized mutual information and all-confidence to perform bi-level pruning. We also identify the redundancy existing in the set of QCPs and propose effective techniques to eliminate the redundancy. Our extensive experiments on both real and synthetic datasets verify the efficiency of *QCoMine* and the quality of the QCPs. The experimental results also justify the effectiveness of our proposed techniques for redundancy elimination. To further demonstrate the usefulness and the quality of QCPs, we study an application of QCPs to classification. We demonstrate that the classifier built on the QCPs achieves higher classification accuracy than the state-of-the-art classifiers built on association rules.

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

General Terms: Algorithms

Additional Key Words and Phrases: Quantitative Databases, Correlated Patterns, Information-Theoretic Approach, Mutual Information

1. INTRODUCTION

Mining correlations [Brin et al. 1997; Motwani et al. 2001; Ma and Hellerstein 2001; Lee et al. 2003; Kim et al. 2004; Xiong et al. 2006; Xiong et al. 2006; Zhang and Feigenbaum 2006] is recognized as an important data mining task for its many advantages over mining association rules [Agrawal et al. 1993a]. Instead of discovering co-occurrence patterns in data as does association rule mining, correlation mining identifies the underlying dependency from the database. More importantly,

Authors' address: Department of Computer Science and Engineering, HKUST, Clear Water Bay, Kowloon, Hong Kong. Emails: {keyiping,csjames,wilfred}@cse.ust.hk

correlation mining does not rely on the *support* measure to quantify the interestingness of the patterns; thus, correlated patterns are not restricted to frequently co-occurring attributes. As a result, 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 (non-commonsense) but important incidents, such as diseases, network intrusions, earthquakes and so on, and their possible causes.

Existing research on correlation mining is primarily conducted on boolean databases [Brin et al. 1997; Ma and Hellerstein 2001; Motwani et al. 2001; Lee et al. 2003; Kim et al. 2004; Xiong et al. 2006; Xiong et al. 2006; Zhang and Feigenbaum 2006]. However, most attributes in real-life databases are not restricted to taking only boolean values. Instead, these attributes can be *quantitative*, which are numeric values (e.g., an employee's salary), and *categorical*, which are enumerations (e.g., different education levels). We refer to databases that consist of quantitative and/or categorical attributes 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 than mining boolean databases from the technical perspective. On the one hand, the stronger expressive power of quantitative attributes over boolean attributes allows us to obtain much richer knowledge than from boolean databases. On the other hand, the expressiveness of quantitative attributes aggravates the complexity of mining quantitative databases due to the large domain size of the attributes.

It may appear that we can apply existing boolean correlation mining algorithms to mine the quantitative correlated patterns, by first discretizing the quantitative attributes and then mapping each discretized interval to a boolean variable. However, this approach has a number of drawbacks. First, discretization has a dramatic impact on the quality of the resultant patterns. To interpret a resultant pattern, each boolean attribute in the pattern should be mapped back to a quantitative attribute with a discretized interval. Once the discretization is done, the intervals of a quantitative attribute that appear in the final pattern are fixed, regardless of whatever other attributes may appear in the same pattern. Second, this approach ignores the dependency between attributes, since a quantitative attribute is discretized and mapped to many boolean attributes. As a result, the dependency between attributes is lost in resultant patterns. Third, the discretization may generate too many intervals, which severely degrades the mining efficiency, since each interval is regarded as an attribute when the boolean correlation mining algorithm is applied. As a result, this approach does not scale well for datasets with a large number of attributes.

There are two combinatorial explosion problems inherent to mining quantitative databases. One is that mining quantitative databases suffers from the combinatorial explosion of the attribute sets, which is similar to mining boolean databases. Given a pattern containing n attributes, the number of its non-empty subsets is $(2^n - 1)$. Thus, the search space of the mining operation can become extremely large. Another more severe combinatorial explosion problem is due to the interval sets arising from different combinations of the values in the large domains of quantitative

attributes. These two combinatorial explosions result in the high complexity of mining quantitative databases and can severely degrade the mining efficiency.

There have been a number of studies on mining association rules from quantitative databases [Srikant and Agrawal 1996; Wang et al. 1998; Fukuda et al. 2001; Aumann and Lindell 2003; Zhang et al. 2004; Ruckert et al. 2004; Chen and Liu 2005; Ke et al. 2006a]. However, as mentioned above, the complexity of mining quantitative association rules is very high due to the combinatorial explosion of frequent patterns, from which association rules are derived. We believe that mining correlated patterns is a more feasible approach to mining quantitative databases, because correlation imposes a stricter definition on the patterns and thus we are able to obtain greater pruning on the search space to reduce the mining complexity. Compared with mining quantitative association rules that may involve a great amount of commonsense and redundant patterns, our approach returns a smaller but more specific set of correlated patterns, which further facilitates advanced analysis.

In this paper, we propose to mine correlations from quantitative databases using an information-theoretic approach. Table I shows an employee database as a running example used throughout this paper. We first illustrate by Example 1 why our approach is desirable.

EXAMPLE 1. The database given in Table I is a relation, r_{emp}, whose schema consists of six attributes: `age`, `education`, `gender`, `marital_status`, `salary` and `service_years`. The quantitative attributes are `age`, `salary` and `service_years`, while other attributes are categorical. The values of each attribute in the original employee relation, r, are now mapped into a set of consecutive integers as given in r_{emp}. The last column records the support value, *supp*, of a transaction T in r_{emp}, meaning that the occurrence probability, p, of those tuples corresponding to T in r.

Table I. An Employee Database, r_{emp}

age	education	gender	marital_status	salary	service_years	$supp(T)$ or $p(T)$
3	2	1	1	1	4	*0.25*
5	1	1	1	2	3	*0.19*
2	2	1	1	2	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*
4	2	1	1	2	1	*0.08*
5	3	2	1	4	3	*0.06*
3	3	2	1	4	2	*0.03*
1	2	2	2	3	2	*0.01*

Consider the *contingency table* [Pearson 1904] in Table II that shows the probabilities of the values of the attributes `gender` and `marital_status`. For quantitative association rule mining, the pattern $\{\texttt{gender} = 1, \texttt{marital\_status} = 1\}$ is a frequent pattern due to its high support of 0.81. However, if we investigate the underlying dependency of `gender` and `marital_status`, we find that the two attributes

Table II. The Contingency Table of Two Attributes, gender and marital_status

gender \ marital_status	1	2	Total
1	**0.81**	0.09	0.9
2	0.09	0.01	0.1
Total	0.9	0.1	1

are totally independent of each other. This can be verified by checking the independency condition that for every possible value of $v_{\texttt{gender}}$ and $v_{\texttt{marital\_status}}$, the equation $p(v_{\texttt{gender}}, v_{\texttt{marital\_status}}) = (p(v_{\texttt{gender}}) \cdot p(v_{\texttt{marital\_status}}))$ holds. For instance, $p(\texttt{gender} = 1,\ \texttt{marital\_status} = 1) = p(\texttt{gender} = 1) \cdot p(\texttt{marital\_status} = 1) = 0.9 \times 0.9 = 0.81$. ■

The above example shows that the *support* measure used by association rule mining is not effective in capturing the true underlying dependency relationship between the attributes. This motivates us to devise a new quality measure of patterns as well as an efficient algorithm to discover the patterns in quantitative databases.

1.1 Contributions

We propose the novel notion of *Quantitative Correlated Pattern* (*QCP*) based on two dependency measures: a newly proposed *Normalized Mutual Information* (*NMI*) founded on information theory [Shannon 1948], and a generalized notion of *all-confidence*, which was originally developed for boolean correlated patterns [Omiecinski 2003; Ma and Hellerstein 2001]. Based on these two dependency measures, we are able to achieve *bi-level* quality control in mining QCPs. First, we employ NMI to specify a required minimum degree of dependency among all attributes in a pattern. Second, we use all-confidence to enforce correlation at a finer granularity on the specific intervals of the quantitative attributes.

Next, we develop an efficient algorithm, *QCoMine*, for mining QCPs. Our algorithm consists of three key components: the *supervised interval combining method*, the *attribute-level pruning* by NMI and the *interval-level pruning* by all-confidence.

In mining quantitative databases, the large domain of a quantitative attribute is first mapped into a number of small intervals. During the mining process, consecutive intervals are combined to gain sufficient support values as well as to produce meaningful intervals [Srikant and Agrawal 1996; Wang et al. 1998]. We develop a *supervised interval combining method* specifically for mining correlations 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 mutual information to guide the interval combining of one attribute with respect to another attribute. We model the interval combining problem as an optimization problem and devise a fast greedy algorithm as a solution.

After processing the intervals of attributes, QCoMine mines the set of QCPs by performing effective bi-level pruning. At the attribute level, we define an *NMI graph* based on the NMI values of the attributes and incorporate the NMI graph into our mining process. By following the NMI graph, we are able to prune an overwhelming number of uncorrelated patterns that are generated from those attributes with

low mutual dependency. At the interval level, all-confidence is applied to further prune the uncorrelated intervals of the highly dependent attributes. Using the *downward closure* and *cross-support* properties [Omiecinski 2003; Xiong et al. 2006], the pruning by all-confidence quickly reduces a large search space to a small one.

We further examine the set of QCPs returned from QCoMine and find that redundancy exists at both the attribute level and the interval level of the patterns. To remove the redundancy, we first define the set of *all-confidence-closed* QCPs, which is a lossless representation of the set of QCPs. Then, we employ the property of all-confidence-closed QCPs to perform effective pruning in mining the set of all-confidence-closed QCPs. At the interval level, we define redundant QCPs based on the proximity of the intervals of QCPs to the combined intervals obtained by our supervised interval combining method. As a result, our approach is able to return a concise and non-redundant set of QCPs.

Finally, our extensive experiments show that the supervised interval combining method and the bi-level pruning by NMI and all-confidence are essential to the efficient mining of quantitative databases. The experimental results verify that the supervised interval combining method not only produces meaningful intervals but also serves as an effective tool that avoids the generation of trivial patterns. By examining the QCPs, we also verify the effectiveness of using NMI and all-confidence in pruning uncorrelated patterns. In addition, we show that our techniques for eliminating the redundant QCPs are effective and efficient, since the set of non-redundant QCPs obtained is up to 20 times smaller than the original set of QCPs, while the incorporation of the elimination of redundancy does not degrade the mining efficiency at all.

We further examine the feasibility of our approach to mining correlated patterns compared with mining frequent patterns from quantitative databases [Srikant and Agrawal 1996]. We find that frequent patterns are mostly patterns with either very low all-confidence (i.e., the patterns are almost uncorrelated) or trivial intervals (i.e., the patterns are just common knowledge), while the majority of the patterns obtained by QCoMine are rare but highly correlated. When mining quantitative frequent patterns becomes infeasible even under very restrictive settings, such as with very large minimum support thresholds, QCoMine still achieves impressive performance.

We further study the quality and usefulness of QCPs in the context of classification. Existing work [Hu et al. 1999; Li et al. 2001; Yin and Han 2003] has studied the use of association rules to predict the class labels of unlabeled data. The classifiers built on association rules are called *associative classifiers*. Associative classifiers are shown to achieve higher classification accuracy than the traditional classifier C4.5 [Quinlan 1993]. In our work, we generate a set of association rules from QCPs and then feed them into an associative classifier. Our experimental results show that the classification accuracy obtained using QCPs is significantly higher than that of three state-of-the-art associative classifiers. The results verify that QCPs can indeed capture the underlying dependency of attributes embedded in the database.

The contributions of this paper are summarized as follows.

—We propose a novel notion of QCP and a new bi-level quality control, both at the coarser attribute level and at the finer interval level, that assures the quality

of the discovered correlated patterns.

—We design an effective supervised interval combining method that is specifically for mining correlations from quantitative databases.

—We develop a bi-level pruning technique, which consists of the attribute-level pruning by NMI and the interval-level pruning by all-confidence. The bi-level pruning is applied to devise an efficient algorithm, QCoMine, for mining QCPs.

—We propose effective and efficient techniques that eliminate the redundancy at the attribute and interval levels of the QCPs.

—We conduct extensive experiments that examine the impacts of the fundamental components in QCoMine and verify the effectiveness and the efficiency of our approach for mining quantitative databases. We also study an application of QCPs for the problem of classification.

1.2 Organization

We present preliminaries in Section 2. We define NMI in Section 3, based on which QCPs are formally defined in Section 4. We present a supervised interval combining method for quantitative attributes in Section 5 and the algorithm for QCP mining, *QCoMine*, in Section 6. We develop the redundancy removal techniques for QCPs in Section 7. Then, we analyze the performance study in Section 8. Finally, we discuss related work in Section 9 and conclude our paper in Section 10.

2. PRELIMINARIES

Let $\mathcal{I} = \{x_1, x_2, \ldots, x_m\}$ be a set of distinct *attributes* or *random variables*[1]. We use $\mathcal{Q}$ and $\mathcal{C}$ to denote the set of *quantitative* attributes and the set of *categorical* attributes, respectively. 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 $x \in \mathcal{C}$, and $l_x \leq u_x$, if $x \in \mathcal{Q}$.

A *quantitative pattern* (or simply called a *pattern*) is a nonempty set of items having 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 the k-attribute set and the k-interval set, where k is the cardinality of the respective set. Given two patterns, X and Y, we say that X is a *sub-pattern* of Y (or Y is a *super-pattern* of X), denoted as $X \subseteq Y$ (or $Y \supseteq X$), if $\forall x[l_x, u_x] \in X \Rightarrow x[l_x, u_x] \in Y$. For brevity, we write the pattern $\{x[l_x, u_x], y[l_y, u_y]\}$ as $x[l_x, u_x]y[l_y, u_y]$.

Given two intervals, $[l, u]$ and $[l', u']$, we say that $[l', u']$ is a *sub-interval* of $[l, u]$ (or $[l, u]$ is a *super-interval* of $[l', u']$), denoted as $[l', u'] \sqsubseteq [l, u]$ (or $[l, u] \sqsupseteq [l', u']$), if $l \leq l' \leq u' \leq u$.

We assume a lexicographic order on the set of attributes, $\mathcal{I}$. Thus, the items and the interval set of a pattern are ordered according to the order of the attribute set of the pattern.

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 that T *supports* a pattern X if $\forall j \in \{1, \ldots, m\}$, $x_j \in attr(X) \Rightarrow l_j \leq v_j \leq$

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

u_j. A *quantitative database* $\mathcal{D}$ is a set of transactions. The *frequency* of a pattern X in $\mathcal{D}$, denoted as $freq(X)$, is the number of transactions in $\mathcal{D}$ that support X. The *support* of X, denoted as $supp(X)$, is the probability that a transaction T in $\mathcal{D}$ supports X, and it is defined as $supp(X) = \frac{freq(X)}{|\mathcal{D}|}$.

EXAMPLE 2. Consider the employee database given in Table I. We have $\mathcal{I} = \{$`age`, `education`, `gender`, `marital_status`, `salary`, `service_years`$\}$, $\mathcal{Q} = \{$`age`, `salary`, `service_years`$\}$, and $\mathcal{C} = \mathcal{I} \setminus \mathcal{Q}$. An example item is `age`$[3, 4]$ and an example 2-pattern is $X =$ `age`$[3, 4]$`education`$[2, 2]$ with the attribute set $\{$`age`, `education`$\}$ and the interval set $\{[3, 4], [2, 2]\}$. Each row in Table I represents a transaction together with its support value in the last column. Since only the first and the seventh transactions support the pattern X, the support value of X is given by adding the support values of these two transactions, i.e., $supp(X) = 0.25 + 0.08 = 0.33$. ■

3. NORMALIZED MUTUAL INFORMATION

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

3.1 Entropy and Mutual Information

Entropy and *Mutual Information* (*MI*) are two central concepts in information theory [Shannon 1948]. Entropy measures the uncertainty of a random variable, while MI describes how much information one random variable tells about another one.

We first define entropy and MI. Table III lists the notation used throughout this paper. Note that $p(v_x)$ is equivalent to $supp(x[v_x, v_x])$, while $p(v_x, v_y)$ is equivalent to $supp(x[v_x, v_x]y[v_y, v_y])$. Thus, these terms are used interchangeably in the subsequent discussions.

Table III. Notations

Notation	Description
x, y, ...	random variables (or attributes)
v_x, v_y, ...	attribute values in their respective domains
$x[l_x, u_x]$, $y[l_y, u_y]$, ...	items corresponding to attributes
X, Y, ...	quantitative patterns (i.e. a set of items)
$p(v_x)$	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 $(x = v_x)$

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. The detailed proof of the following properties can be consulted from [Cover and Thomas 1991].

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

Property 1 gives an important interpretation of MI in terms of entropy. Intuitively, 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, the entropy of a variable is also called the *self-information* of the variable.

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 value of v_x and v_y, which means that x and y are independent. Informally, this means that 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, which is the minimum of $H(x)$ and $H(y)$.

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

3.2 Normalized Mutual Information

Although MI serves as a good measure to quantify how closely two attributes are related to each other, the MI values of the attributes do not conform to a unified scale. As shown by Properties 4 and 5, the MI value of two attributes varies between 0 and the minimum of their entropy. Since the entropy of different attributes often varies a lot, different pairs of attributes also have different ranges of MI values, which is not desirable as a dependency measure. In order to make MI values mutually comparable in our mining problem, we need a unified scale for measuring MI among a global set of attributes $\mathcal{I}$. We thus propose the concept of normalized MI.

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:*

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

When both $I(x;x)$ and $I(y;y)$ are zero, we define $\widetilde{I}(x;y) = 0$.

The underlying idea of Definition 3 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 Properties 3 and 5. As a result, we eliminate the localness of an attribute pair and use NMI as a global measure of attribute dependency. Properties 7 to 9 present some important properties of NMI.

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

Proof. This property follows directly from Property 2. □

Property 7 shows that NMI is also symmetric, which preserves the nice feature of MI.

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 Property 5, $I(x;y) \leq MIN\{H(x), H(y)\} \leq MAX\{H(x), H(y)\}$. By Property 3, it follows that $I(x;y) \leq MAX\{I(x;x), I(y;y)\}$. So $\widetilde{I}(x;y) \leq 1$. □

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 formalizes the semantics of NMI, which is *the minimum fraction of reduction in the uncertainty of one attribute given the knowledge of another attribute*.

Apart from the above stated properties, NMI serves as a natural measure of correlation for the following reasons. First, NMI is a formal concept for measuring dependency between attributes. Second, NMI gives an intuitive meaning for quantifying the degree of dependency: NMI takes a value of 0 to indicate independence and the value of NMI increases within the unit range $[0, 1]$ when the dependency of attributes increases. Third, a threshold μ for NMI can be employed to indicate the required minimum fraction of reduction in the uncertainty of an attribute given the knowledge of another attribute.

EXAMPLE 3. Given the employee database in Table I, by Definition 2, we have

$$\begin{aligned} & I(\texttt{age};\ \texttt{marital\_status}) \\ = & \sum_{v_{\texttt{age}} \in \{1,2,3,4,5\}} \sum_{v_{\texttt{marital\_status}} \in \{1,2\}} p(v_{\texttt{age}},\ v_{\texttt{marital\_status}}) \cdot \log \frac{p(v_{\texttt{age}},\ v_{\texttt{marital\_status}})}{p(v_{\texttt{age}})p(v_{\texttt{marital\_status}})} \\ = & \ 0.47. \end{aligned}$$

This shows that the knowledge of `age` (or `marital_status`) causes a reduction of 0.47 in the uncertainty of `marital_status` (or `age`). However, it is difficult to tell how much a reduction of 0.47 is. However, using NMI, we obtain

$$\begin{aligned} & \widetilde{I}(\texttt{age};\ \texttt{marital\_status}) \\ = & \frac{I(\texttt{age};\ \texttt{marital\_status})}{MAX\{H(\texttt{age}),\ H(\texttt{marital\_status})\}} \\ = & \frac{I(\texttt{age};\ \texttt{marital\_status})}{H(\texttt{age})} \\ = & \frac{0.47}{2.12} \\ = & \ 0.22. \end{aligned}$$

Thus, there is a reduction of at least 22% of the uncertainty of `age` and `marital_status`.

Similarly, we obtain $I(\texttt{education};\ \texttt{gender}) = 0.40$ and $\widetilde{I}(\texttt{education};\ \texttt{gender}) = 0.30$.

Note that we have $I(\texttt{age};\ \texttt{marital\_status}) > I(\texttt{education};\ \texttt{gender})$ but $\widetilde{I}(\texttt{age};\ \texttt{marital\_status}) < \widetilde{I}(\texttt{education};\ \texttt{gender})$. The larger MI value of `age` and `marital_status` is mainly because the entropy of `age` is larger than that of `education` (i.e., $H(\texttt{age}) = 2.12 > H(\texttt{education}) = 1.32$), which results in a larger absolute amount of reduction in the uncertainty rather than a relative amount. This illustrates the benefit of NMI, which is able to reflect the attribute dependency better than MI does. ■

In addition to the above definition of NMI, we can also define the normalization as follows:

—$\frac{I(x;y)}{MIN\{I(x;x),I(y;y)\}}$: this expression measures the maximum fraction of reduction in the uncertainty of one attribute given the knowledge of another attribute.

—$\frac{2I(x;y)}{I(x;x)+I(y;y)}$: this expression measures the average fraction of reduction in the uncertainty of one attribute given the knowledge of another attribute.

The above two alternatives also possess Properties 7 and 8, i.e., they are symmetric and their values fall within $[0, 1]$. However, they are weaker than the one defined in Definition 3 as a measure of the quality of a correlated pattern.

4. QUANTITATIVE CORRELATED PATTERNS

In this section, we present the concept of *Quantitative Correlated Pattern* (QCP), which employs *bi-level* quality control on the correlated patterns to be mined. First, NMI is applied as a dependency measure at the attribute level to identify highly correlated attributes. Then, all-confidence is applied at the interval level to ensure the correlation of the attributes with specific intervals. In the previous section, we have discussed NMI. Now, we present the concept of all-confidence generalized for quantitative patterns and define the notion of QCP, which is the core concept in our work.

4.1 All-Confidence for Quantitative Patterns

There have been a number of proposals [Brin et al. 1997; Omiecinski 2003; Ma and Hellerstein 2001; Tan et al. 2002; Xiong et al. 2006] for measuring correlation relationships. In recent years, *all-confidence* has emerged as a commonly adopted correlation measure. It has been shown in many studies [Omiecinski 2003; Ma and Hellerstein 2001; Lee et al. 2003; Kim et al. 2004; Xiong et al. 2006] that all-confidence reflects correlative relationships 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 $allconf(X)$, is defined as:*

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

A pattern that has all-confidence of no less than a given *minimum all-confidence threshold*, ς, indicates a high correlation among all the items in the pattern. This is because *all* association rules derived from the pattern have confidence of no less than ς, as implied by Definition 4. Note that an association rule is only a one-way implication from the set of items on the left-hand side of the rule to that on the right-hand side.

EXAMPLE 4. Given the employee database in Table I and the pattern $X = \texttt{gender}[1,1]\texttt{salary}[2,2]$, we have

$$\begin{aligned} & allconf(X) \\ &= \frac{supp(\texttt{gender}[1,1]\texttt{salary}[2,2])}{MAX\{supp(\texttt{gender}[1,1]),\ supp(\texttt{salary}[2,2])\}} \\ &= \frac{0.19+0.11+0.09+0.09+0.08}{MAX\{0.25+0.19+0.11+0.09+0.09+0.09+0.08,\ 0.19+0.11+0.09+0.09+0.08\}} \\ &= 0.62. \end{aligned}$$

Similarly, we can compute the all-confidence of the pattern $Y = \texttt{gender}[1,1]$ $\texttt{marital\_status}[1,1]$ to be $allconf(Y) = 0.9$, which indicates a higher correlation among its items than that among the items of X. ■

All-confidence has two desirable properties for the efficient mining of correlated patterns from boolean databases. These two properties can be directly adapted as effective pruning tools for mining quantitative patterns, since the sub-pattern in a quantitative database is defined in the same way as that in a boolean database.

The first property is called the *downward closure property* [Omiecinski 2003], which is formally stated as follows.

PROPERTY 10. (DOWNWARD CLOSURE PROPERTY OF ALL-CONFIDENCE) *Given two patterns, X and Y, if $X \subseteq Y$, then $allconf(X) \geq allconf(Y)$.*

By Property 10, we are able to perform the following pruning: if a pattern X has an all-confidence value less than ς, then we can prune all its super-patterns.

The second property is called the *cross-support property* [Xiong et al. 2006], which is stated as follows.

PROPERTY 11. (CROSS-SUPPORT PROPERTY OF ALL-CONFIDENCE) *Given a pattern X, if there exist two items, $x[l_x, u_x]$, $y[l_y, u_y] \in X$ such that $\frac{supp(x[l_x,u_x])}{supp(y[l_y,u_y])} < \varsigma$, then $allconf(X) < \varsigma$.*

By Property 11, if the support ratio between any two items in a pattern X is less than ς, we can prune the pattern X without computing its all-confidence value.

Although all-confidence is a good measure of correlation among boolean attributes, it is still 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 the attributes. However, quantitative attributes often consist of a large number of intervals; thus, we may obtain patterns that have high all-confidence simply as a result of co-occurrence. We will illustrate this limitation further in Example 5. On the other hand, although NMI is a good measure of dependency among attributes, it does not capture the specific relationships between the attribute values, since it aggregates all the information among the attribute values into a single NMI value. As a result, the patterns measured only by NMI are

not specific enough to provide detailed knowledge to the end user. In order to identify correlations at different levels, we integrate both measures into the definition of *quantitative correlated patterns*, as will be detailed in the following subsection.

4.2 Quantitative Correlated Patterns

Realizing that the definition of a correlated pattern [Brin et al. 1997] is *a set of attributes that are dependent on each other* and that MI is a well-established concept in information theory [Shannon 1948] to *capture the dependency among attributes*, we formally incorporate the concept of MI into the definition of QCP.

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

The essence of the above definition is that we first ensure that every attribute in a QCP carries a great amount of information about every other attribute in the pattern. Then, all-confidence is further used to guarantee that the intervals of the attributes are also highly correlated. Therefore, bi-level quality control is imposed on the QCPs.

The following example helps illustrate further the concept of QCP, as well as the necessity of imposing the bi-level quality control for the QCPs.

EXAMPLE 5. We refer to the employee database in Table I. Let $\mu = 0.3$ and $\varsigma = 0.6$. The pattern $X = \texttt{gender}[1,1]\texttt{salary}[2,2]$ is a QCP, since $\widetilde{I}(\texttt{gender}; \texttt{salary}) = 0.34 \geq \mu$, and $allconf(X) = 0.62 \geq \varsigma$ as shown in Example 4.

The pattern $Y = \texttt{gender}[1,1]\texttt{marital\_status}[1,1]$ is not a QCP because $\widetilde{I}(\texttt{gender}; \texttt{marital\_status}) = 0 < \mu$, even though $allconf(Y) = 0.9 \geq \varsigma$. As shown in Example 1, the attributes `gender` and `marital_status` are actually independent of each other. The reason for the high all-confidence of Y is simply because both $p(\texttt{gender}[1,1])$ and $p(\texttt{marital\_status}[1,1])$ are very high (both of them are 0.9 as shown in Table II), which results in a high co-occurrence of the two items $\texttt{gender}[1,1]$ and $\texttt{marital\_status}[1,1]$. Obviously, patterns such as Y are of little significance, since they do not accurately reveal the correlations between the items in the patterns. This explains the necessity of the concept of NMI in the definition of QCP. ■

We now define the QCP mining problem that we tackle in this paper.

Problem Description. Given a quantitative database $\mathcal{D}$, a minimum information threshold μ, and a minimum all-confidence threshold ς, the *QCP mining problem* is to find all QCPs from $\mathcal{D}$.

5. A SUPERVISED INTERVAL COMBINING METHOD

In this section, we present a supervised interval combining method for quantitative attributes, which is an essential technique to produce meaningful intervals for QCPs,

as well as to ensure the efficient mining of QCPs.

A very important step in mining quantitative databases is the process of producing meaningful intervals of the attributes. To deal with continuous values, the quantitative database is first discretized so that the domain of each quantitative attribute is mapped into a set of *base intervals*. The base intervals are indivisible during the mining process. We discretize the domain of each quantitative attribute into a relatively large number of base intervals so that the information loss is small. Consecutive base intervals may be combined into larger intervals to gain sufficient support, while a combined interval itself can have a more significant meaning than its composite base intervals. However, the challenge is how to avoid producing the combined intervals that are 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 [Srikant and Agrawal 1996] is not applicable to the QCP mining problem. This is because QCPs can be both rare patterns (having low support) and popular patterns (having high support) and thus they have a wide range of support values. Other more sophisticated interval combining methods, such as [Wang et al. 1998], have also been proposed but are specifically designed for mining quantitative association rules.

In mining QCPs, it would be advantageous to consider the dependency between the attributes when combining their intervals to reflect specific meanings. For example, combining the intervals of the attribute `age` with respect to `marital_status` (as people are married after a certain age) should be different from that with respect to `gender` (as both males and females scatter over all ages). However, most of the existing interval combining methods produce a set of intervals for an attribute without considering the relationship of the attribute with other attributes. As a result, no matter what attributes appear in the same pattern, the intervals produced for a specific attribute are the same. To address this problem, we propose a novel interval combining method, in which the intervals of an attribute are combined with respect to other attributes in a supervised process.

Since interval combining is performed locally between a pair of attributes, we use MI to guide the process to produce meaningful combined intervals, instead of NMI, which is defined to be a global measure for all attributes. We now model the interval combining problem as a supervised optimization problem with MI as the objective function.

Given two attributes x and y, where x is quantitative and y is either categorical or quantitative, we aim to obtain the optimal combined intervals of x with respect to y. If y is also quantitative, we obtain the combined intervals of y with respect to x as well. Let $I'(x;y)$, $H'(x)$, $H'(y)$ and $H'(x,y)$ be the respective parameters after combining the intervals of x (and y). The objective function, ϕ, of the optimization problem is defined as follows:

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

Note that if y is categorical, $H(y)$ remains unchanged, because the intervals of y are not combined.

Since $H(x)$, $H(y)$ and $H(x,y)$ always decrease when the intervals of x (and y) are combined, ϕ can be either positive or negative, depending on the relative decreasing rate of $H(x)$, $H(y)$ and $H(x,y)$. Thus, the optimization problem is to maximize the function ϕ. Obviously, an exhaustive algorithm is unrealistic, since it requires $\mathcal{O}(2^{n_x+n_y})$ computations of MI values to find the optimal solution, where n_x and n_y are the numbers of the base intervals of x and y. We propose a greedy algorithm as a solution to this optimization problem based on the following property of the objective function ϕ.

Let $\mathcal{S}$ be the set of intervals of x and y produced at a certain interval combining step, and $\phi_{\mathcal{S}}(x,y)$ be the corresponding value of $\phi(x,y)$. Let $\mathcal{T}$ be the set of intervals of x and y obtained by further combining some intervals in $\mathcal{S}$, and $\phi_{\mathcal{T}}(x,y)$ be the corresponding value of $\phi(x,y)$. Since $\mathcal{T}$ can be derived from $\mathcal{S}$, we use $\phi_{\mathcal{S}\to\mathcal{T}}(x,y)$ to denote the value of $\phi(x,y)$ of obtaining $\mathcal{T}$ from $\mathcal{S}$. We present the additive property of ϕ as follows.

PROPERTY 12. (ADDITIVE PROPERTY) $\phi_{\mathcal{T}}(x,y) = \phi_{\mathcal{S}}(x,y) + \phi_{\mathcal{S}\to\mathcal{T}}(x,y)$.

PROOF. We use the subscripts $\mathcal{S}$ and $\mathcal{T}$ to denote the corresponding values of MI.

$$\begin{aligned}\phi_{\mathcal{S}}(x,y) + \phi_{\mathcal{S}\to\mathcal{T}}(x,y) &= (I_{\mathcal{S}}(x;y) - I(x;y)) + (I_{\mathcal{T}}(x;y) - I_{\mathcal{S}}(x;y))\\ &= I_{\mathcal{T}}(x;y) - I(x;y)\\ &= \phi_{\mathcal{T}}(x,y).\end{aligned}$$

□

Property 12 shows that the order of combining the intervals does not affect the value of ϕ as long as the final set of combined intervals is the same. Moreover, it also shows that, we can obtain the value of ϕ by summing up the intermediate values of ϕ, instead of recomputing it from scratch at each combining step.

Based on Property 12, we propose an efficient algorithm that greedily combines two consecutive intervals of a given attribute at each time. The idea of the greedy algorithm is described as follows.

At each time, we consider combining two consecutive intervals of x (or y, if $y \in \mathcal{Q}$). Without loss of generality, we assume that the two intervals to be combined are i_{x_j} and $i_{x_{j+1}}$, of x, where i_{x_j} and $i_{x_{j+1}}$ can be either a base interval or a combined interval. Let $\phi_{[i_{x_j}, i_{x_{j+1}}]}(x,y)$ denote the value of $\phi(x,y)$ when i_{x_j} and $i_{x_{j+1}}$ are combined with respect to y. Using Equation (1) and Definition 1, we obtain $\phi_{[i_{x_j}, i_{x_{j+1}}]}(x,y)$ as follows:

$$
\begin{aligned}
& \phi_{[i_{x_j}, i_{x_{j+1}}]}(x, y) \\
= \; & (H'(x) - H(x)) + (H'(y) - H(y)) - (H'(x, y) - H(x, y)) \\
= \; & \Big(- (p(i_{x_j}) + p(i_{x_{j+1}})) \log(p(i_{x_j}) + p(i_{x_{j+1}})) \\
& + (p(i_{x_j}) \log p(i_{x_j}) + p(i_{x_{j+1}}) \log p(i_{x_{j+1}})) \Big) + \; 0 \\
& - \Big(- \sum_{i_y} (p(i_{x_j}, i_y) + p(i_{x_{j+1}}, i_y)) \log(p(i_{x_j}, i_y) + p(i_{x_{j+1}}, i_y)) \\
& + \sum_{i_y} p(i_{x_j}, i_y) \log p(i_{x_j}, i_y) + \sum_{i_y} p(i_{x_{j+1}}, i_y) \log p(i_{x_{j+1}}, i_y) \Big).
\end{aligned}
$$

Our algorithm, *GreedyCombine*, is presented as Algorithm 1. The main idea (Steps 21-25) 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 $\phi_{[i_{x_{j'}}, i_{x_{j+2}}]}(x, y)$. If y is quantitative (Steps 4-19), we also take into account the values of ϕ for combining the consecutive intervals of y. At each time, we also pick up the maximum $\phi_{[i_{y_k}, i_{y_{k+1}}]}(x, y)$ among all pairs of consecutive intervals, i_{y_k} and $i_{y_{k+1}}$ of y (Step 9) and determine the intervals of which attribute to be combined by comparing the two maximum values of ϕ of x and y (Steps 11 and 16).

For each quantitative attribute, we maintain a *heap* to achieve the efficient retrieval of maximum value of ϕ. Take the attribute x for example. We can retrieve the maximum $\phi_{[i_{x_j}, i_{x_{j+1}}]}(x, y)$ by implementing a priority queue using a heap Q_x (Step 2), 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 keeping their pointers in the heap entry of $\phi_{[i_{x_j}, i_{x_{j+1}}]}(x, y)$. The update of their positions in Q_x by a *heapify* operation takes $\mathcal{O}(\log n_x)$ time (Steps 13 and 18). If y is quantitative, the update of the values of ϕ in Q_y when combining two intervals of x takes n_y *heapify* operations (Step 13). However, we can simply perform *build-heap* to rebuild Q_y which takes only $\mathcal{O}(n_y)$ time. Moreover, both n_x and n_y decrease when more intervals are combined. In the worst case, when all intervals of both x and y are combined into a single interval, the entire combining process takes n_x *heapify* operations for Q_x and n_y for Q_y, n_y *build-heap* operations for Q_x and n_x for Q_y. Thus, the total complexity in the worst case is $\mathcal{O}(n_x \log n_x + n_y \log n_y + n_x \cdot n_y)$.

To avoid a combined interval becoming trivial, we set a terminating condition, ϕ^x_{min}, to be the mean of all $\phi_{[i_{x_j}, i_{x_{j+1}}]}(x, y)$ in the heap Q_x (Step 3). This initial value of ϕ^x_{min} is chosen so that those pairs of consecutive intervals with relatively high ϕ values are given a chance to be combined. When intervals are combined, the heap Q_x is updated (Steps 13 and 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.

In the same way, we also set a terminating parameter, ϕ^y_{min}, for the attribute y if y is quantitative. The *GreedyCombine* terminates when the intervals of both attributes cannot be combined any more with respect to their respective ϕ_{min}.

Algorithm 1 *GreedyCombine*(x, y)

Input: The base intervals of two attributes x and y.
Output: The set of combined intervals of x and y.

1. **for each** pair of consecutive intervals i_{x_j} and $i_{x_{j+1}}$ of x **do**
2. Insert $\phi_{[i_{x_j}, i_{x_{j+1}}]}(x, y)$ into a heap, Q_x;
3. $\phi^x_{min} \leftarrow MEAN\{\phi_{[i_{x_j}, i_{x_{j+1}}]}(x, y) \in Q_x\}$;
4. **if** $(y \in \mathcal{Q})$
5. **for each** pair of consecutive intervals i_{y_k} and $i_{y_{k+1}}$ of y **do**
6. Insert $\phi_{[i_{y_k}, i_{y_{k+1}}]}(x, y)$ into a heap, Q_y;
7. $\phi^y_{min} \leftarrow MEAN\{\phi_{[i_{y_k}, i_{y_{k+1}}]}(x, y) \in Q_y\}$;
8. Extract maximum $\phi_{[i_{x_j}, i_{x_{j+1}}]}(x, y)$ from Q_x;
9. Extract maximum $\phi_{[i_{y_k}, i_{y_{k+1}}]}(x, y)$ from Q_y;
10. **if** $(\phi_{[i_{x_j}, i_{x_{j+1}}]}(x, y) \geq \phi^x_{min})$
11. **if** $(\phi_{[i_{x_j}, i_{x_{j+1}}]}(x, y) \geq \phi_{[i_{y_k}, i_{y_{k+1}}]}(x, y)$ or $\phi_{[i_{y_k}, i_{y_{k+1}}]}(x, y) < \phi^y_{min})$
12. Combine i_{x_j} and $i_{x_{j+1}}$ into $i_{x_{j'}}$;
13. Update Q_x and Q_y;
14. Goto Step 8;
15. **if** $(\phi_{[i_{y_k}, i_{y_{k+1}}]}(x, y) \geq \phi^y_{min})$
16. **if** $(\phi_{[i_{y_k}, i_{y_{k+1}}]}(x, y) > \phi_{[i_{x_j}, i_{x_{j+1}}]}(x, y)$ or $\phi_{[i_{x_j}, i_{x_{j+1}}]}(x, y) < \phi^x_{min})$
17. Combine i_{y_k} and $i_{y_{k+1}}$ into $i_{y_{k'}}$;
18. Update Q_y and Q_x;
19. Goto Step 8;
20. **else** \\ *y is categorical*
21. Extract maximum $\phi_{[i_{x_j}, i_{x_{j+1}}]}(x, y)$ from Q_x;
22. **if** $(\phi_{[i_{x_j}, i_{x_{j+1}}]}(x, y) \geq \phi^x_{min})$
23. Combine i_{x_j} and $i_{x_{j+1}}$ into $i_{x_{j'}}$;
24. Update Q_x;
25. Goto Step 21;

Example 6. Consider the employee database in Table I, where each label of quantitative attributes corresponds to one base interval. Using *GreedyCombine*, the combined intervals of `age` with respect to `marital_status` are $[1, 1]$ and $[2, 5]$. This is reasonable, since all the transactions with `marital_status` $= 2$ have a value of 1 for `age`. In other transactions with `marital_status` $= 1$, the values of `age` fall within the interval $[2, 5]$. This result is consistent with the fact that people over a certain age, say 35, are usually 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 `marital_status`. Fewer base intervals of `age` are combined with respect to `gender`, since in the transactions with the same `gender` values, the `age` values scatter over the whole domain of `age`. This result demonstrates the fact that there are young, middle-aged and old employees who are either male or female. ■

6. MINING QUANTITATIVE CORRELATED PATTERNS

In this section, we present our algorithm of mining QCPs. Our algorithm utilizes bi-level pruning, which significantly reduces the search space of the QCP 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 the QCP definition (recall Definition 5) requires that, in order to generate a QCP 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 QCP mining problem. By introducing the concept of NMI graphs, we show how the pruning is performed.

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.*

Note that E is well defined in Definition 6, since $\widetilde{I}(x_i; x_j)$ is symmetric by Property 7. We now establish a necessary condition that the attribute set of a QCP forms a *clique* [Cormen et al. 2001] in the NMI graph. The following condition reveals the strong inter-dependence between all attributes in a QCP.

LEMMA 1. (NECESSARY CONDITION) *If* X *is a QCP, then attr(*X*) forms a clique in* G*.*

PROOF. This follows directly from Definitions 5 and 6. □

Lemma 1 implies that we can generate the attribute sets of all QCPs by enumerating the cliques in the NMI graph. Note that mining QCPs without pruning at the attribute level is equivalent to enumerating all cliques in a complete graph; that is, an NMI graph with $\mu = 0$. Thus, the search space is now significantly 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 uncovered if we realize the fact 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 further illustrate this point in Section 6.2.

The complexity of enumerating all cliques in a graph is exponential. However, we show that 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 labeled as 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, the set of nodes, $\{u_1, \ldots, u_k\}$, forms a k-clique in G.*

PROOF. We prove the lemma by induction on the length of the path, k.

(Basis.) When $k = 1$ and $k = 2$, it is trivial that $\{u_1\}$ is a 1-clique and $\{u_1, u_2\}$ forms a 2-clique since (u_1, u_2) is an edge in G.

(Induction.) Assume that the lemma holds for $2 \leq j \leq k$. Consider a path $P_{k+1} = \langle u_1, \ldots, u_{k-1}, u_k, u_{k+1} \rangle$. Thus, u_k must have a right sibling u_{k+1} in T_{attr} and the edge (u_k, u_{k+1}) exists in G. By the inductive hypothesis, let $P_k = \langle u_1, \ldots, u_{k-1}, u_k \rangle$ and $P'_k = \langle u_1, \ldots, u_{k-1}, u_{k+1} \rangle$ be two paths in T_{attr}, and the sets of nodes on P_k and P'_k form two k-cliques. It follows that $\forall u \in \{u_1, \ldots, u_{k-1}\}$ and $\forall v \in \{u_1, \ldots, u_{k-1}, u_k, u_{k+1}\}$, the edge (u, v) exists in G. The result thus follows, since the edge (u_k, u_{k+1}) also exists, giving a $(k+1)$-clique, $\{u_1, \ldots, u_{k-1}, u_k, u_{k+1}\}$, which is the same set of nodes on the path P_{k+1}. □

The prefix tree is shown to be a very efficient data structure for mining both frequent and correlated patterns, while Lemma 2 suggests 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. More importantly, clique enumeration often terminates earlier by all-confidence pruning, which is detailed in Section 6.2.

EXAMPLE 7. Given the employee database in Table I. We compute the NMI graph, G, as shown in Figure 1, at $\mu = 0.3$. There are only five edges in G, each of which is identified as a strong dependency between two attributes. Other edges that cannot constitute a QCP do not exist in G. This ensures that the uncorrelated patterns, such as `gender`$[1, 1]$`marital_status`$[1, 1]$ in Example 5, will not be generated, since there is no edge between `gender` and `marital_status` 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. ■

We now discuss a straightforward and objective way of setting the minimum information threshold μ. As shown in Equation (2), we set μ to be the sum of the *mean*, denoted as *MEAN*, and the *standard deviation*, denoted as *STD*, of all NMI values (the NMI values of $\widetilde{I}(x; x)$ are excluded), so that G retains edges that reveal high mutual dependency between their 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 [Agrawal et al. 1993a], the choice of μ can also be determined by domain experts to indicate how correlated the attributes in a pattern should be.

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

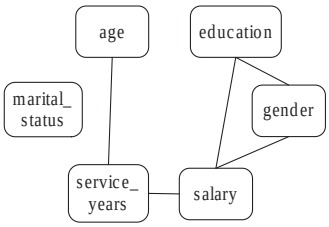


Fig. 1. An NMI Graph, G

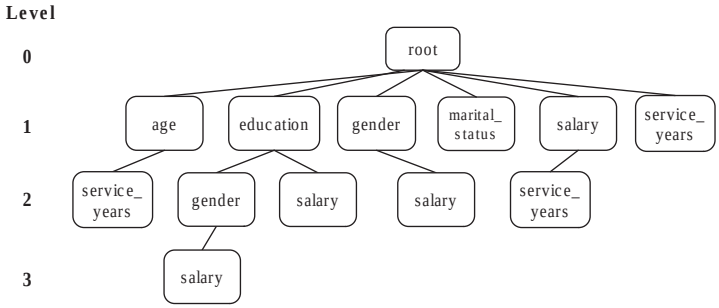


Fig. 2. An Attribute Prefix Tree, T_{attr}

6.2 Interval-Level Pruning

Although NMI can effectively eliminate the patterns that are generated 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 has the same attribute set but different interval sets. Thus, we also need pruning at the interval level. For this purpose, we make use of the downward closure property [Omiecinski 2003] and the cross-support property [Xiong et al. 2006] of all-confidence to perform the pruning.

6.2.1 *Pruning by the Downward Closure Property of All-Confidence.*

According to the downward closure property of all-confidence as stated in Property 10, we can 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 sets 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, a straightforward 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 two patterns $\texttt{age}[30,40]\texttt{marital\_status}[1,1]$ and $\texttt{age}[25,35]\texttt{salary}[2000,3000]$, we intersect the intervals of `age` to obtain a new pattern $\texttt{age}[30,35]\texttt{marital\_status}[1,1]\texttt{salary}[2000,3000]$.

However, producing a $(k+1)$-pattern by intersecting the intervals of k-patterns violates the downward closure property of all-confidence. The reason for the viola-

tion is that 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. The following example helps illustrate this problem.

EXAMPLE 8. Given the employee database in Table I and two 2-patterns $X =$ `education`$[3,3]$`service_years`$[2,3]$ and $Y =$ `gender`$[2,2]$`service_years`$[1,2]$, we can compute

$$allconf(X) = \frac{supp(X)}{supp(\texttt{service\_years}[2,3])} = \frac{0.09}{0.49} = 0.18$$

and

$$allconf(Y) = \frac{supp(Y)}{supp(\texttt{service\_years}[1,2])} = \frac{0.04}{0.3} = 0.13.$$

However, when we generate the 3-pattern, $Z =$ `education`$[3,3]$`gender`$[2,2]$`service_years`$[2,2]$, from X and Y by intersecting the intervals of `service_years`, we find that

$$allconf(Z) = \frac{supp(Z)}{supp(\texttt{gender}[2,2])} = \frac{0.03}{0.1} = 0.3,$$

which is larger than both $allconf(X)$ and $allconf(Y)$. This violates the downward closure property of all-confidence, since the generated pattern has a larger all-confidence value than its composite patterns. The main cause of the violation is that, after the intersection, the maximum support of the single item in the pattern decreases from 0.49 in X and 0.3 in Y to 0.1 in Z. ■

The violation of the downward closure property of all-confidence when intersecting the intervals causes two problems in the mining process. First, using the prefix tree structure introduced in Section 6.1, we are not able to obtain the all-confidence value of the pattern generated by the interval intersection, since the support values of the patterns with the intersected interval are not kept at the upper level of the prefix tree. Second, we cannot discard a pattern X if its all-confidence is less than ς, since after the intersection, X may produce some pattern with an all-confidence value larger than ς. This can be easily shown with Example 8 by setting $\varsigma = 0.2$, in which we have $allconf(X) = 0.18 < 0.2$, but we cannot prune X. Otherwise, we will miss the pattern Z.

Pruning by the downward closure property is well known to be the key to the efficiency of most frequent pattern and correlated pattern mining algorithms [Agrawal et al. 1993a; Omiecinski 2003]. Therefore, the mining efficiency will be severely degraded if the downward closure property of all-confidence cannot be applied in our problem.

Fortunately, we find that the violation of the downward closure property of all-confidence can be resolved by enumerating all sub-intervals of a combined interval before we start to generate a pattern. Hereafter, we use i_x to denote an interval of x (i.e., $i_x = [l_x, u_x]$). For clarity, we also use $x[i_x]$ to denote an item $x[l_x, u_x]$.

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 sets of combined intervals of x and

y with respect to each other. Similar to mining quantitative frequent patterns [Srikant and Agrawal 1996], 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 large, since an interval $i_x = [l, l+n]$ has $\frac{n(n+1)}{2}$ sub-intervals. However, 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 is preserved and can be applied to performing the pruning. The sub-intervals are not intersected and are regarded as indivisible intervals in the mining process.

The set of k-patterns generated at a node at Level k ($k \geq 2$) of T_{attr} 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 avoids storing the duplicate sub-intervals in memory, but it also significantly speeds up the join of two k-patterns to produce a $(k+1)$-pattern.

6.2.2 *Pruning by the Cross-Support Property of All-Confidence.*

According to the cross-support property of all-confidence as stated in Property 11, we can prune a pattern X, if it contains two items whose support ratio is less than ς.

We first apply this property to the generation of 2-patterns. When enumerating all the sub-intervals of x and y, for a sub-interval i'_x of a combined interval i_x of x, we can prune the sub-interval i'_y of y, if $supp(y[i'_y]) < \varsigma \cdot supp(x[i'_x])$ or $supp(y[i'_y]) > \frac{supp(x[i'_x])}{\varsigma}$. In these two cases, the pattern $x[i'_x]y[i'_y]$ is guaranteed to have all-confidence of less than ς according to the cross-support property. Thus, it is not a QCP and can be pruned.

We further apply the property to the generation of k-patterns for $k > 2$. We keep the maximum support value of the single items for each pattern. Given two k-patterns to be joined into a $(k+1)$-pattern, if the ratio of the support value of one pattern to the maximum support value kept for another pattern is less than ς, we do not need to perform the join operation, since the all-confidence value of the joined pattern is guaranteed to be less than ς by the cross-support property.

6.3 QCoMine Algorithm

We now present our main algorithm, *QCoMine*, in Algorithm 2. We first construct the NMI graph G (Step 1). Then, we combine the base intervals of each quantitative attribute with respect to another attribute, as long as the two attributes form an edge in G (Steps 2-3). The attribute prefix tree, T_{attr}, is then constructed to perform the attribute-level pruning. After creating the root and Level 1 of

T_{attr} (Steps 4-5), Steps 6-15 construct Level 2 of T_{attr} and produce all 2-QCPs. Step 12 performs pruning at the interval level by the cross-support property of all-confidence, while the pruning by the downward closure property is performed throughout the mining process.

Steps 16-17 invoke *RecurMine*, as shown in Procedure 1, 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. This ensures that the last interval is a sub-interval of one attribute with respect to another. Step 8 of *RecurMine* performs the pruning by the cross-support property of all-confidence for k-pattern generation ($k > 2$).

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

Input: A quantitative database $\mathcal{D}$, a minimum information threshold μ, and a minimum all-confidence threshold ς.
Output: The set of QCPs.

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

To compute the all-confidence of a pattern, we adopt *diffset* [Zaki and Gouda 2003] to obtain the support value of the pattern, while we use an extra field to keep the maximum support value of the items in a pattern (Denoted as $maxsupp(X)$ for a pattern X in Step 8 of Procedure 1). The use of *diffset*, together with the depth-first strategy, effectively controls the memory consumed in the mining process as evidenced by our experiments discussed in Section 8.

EXAMPLE 9. (Example 7 continued) Let $\mu = 0.3$ and $\varsigma = 0.6$. The corresponding T_{attr} constructed by *QCoMine* is shown in Figure 2. The node `gender` at Level 2 of

Procedure 1 *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 $X = x_1[i_{u_1}] \cdots x_{k-1}[i_{u_{k-1}}]x_k[i_{u_k}]$ and $Y = x_1[i_{u_1}] \cdots x_{k-1}[i_{u_{k-1}}]x_{k+1}[i_{v_k}]$;
8. **if** $(\frac{supp(X)}{maxsupp(Y)} \geq \varsigma$ and $\frac{supp(Y)}{maxsupp(X)} \geq \varsigma)$
9. Let $Z = x_1[i_{u_1}] \cdots x_{k-1}[i_{u_{k-1}}]x_k[i_{u_k}]x_{k+1}[i_{v_k}]$;
10. **if** $(allconf(Z) \geq \varsigma)$
11. Output Z as a QCP;
12. Insert $\{i_{u_1}, \ldots, i_{u_{k-1}}, i_{u_k}, i_{v_k}\}$ into $T^w_{interval}$;
13. Delete $T^u_{interval}$;
14. **for each** child node w of u **do**
15. *RecurMine*$(w, T_{attr}, G, k+1)$;

T_{attr} represents the 2-attribute set {`education`, `gender`}. Since both `education` and `gender` are categorical, 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, the pruning by the cross-support property of all-confidence discards four patterns. Among the remaining two patterns, only the pattern `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 the following five sub-intervals: $[1,1], [2,2], [2,3], [3,3], [4,4]$. Combined with the three values of `education`, there are altogether fifteen sub-interval pairs formed for `education` and `salary`, among which eight corresponding patterns are pruned by the cross-support property, and only one corresponding pattern, `education`$[3,3]$`salary`$[4,4]$, satisfies the all-confidence condition.

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 an all-confidence value of 0.9. ■

7. REDUNDANCY ELIMINATION

Previous studies have already recognized that redundant information exists at the interval level of quantitative association rules [Srikant and Agrawal 1996; Aumann and Lindell 2003] and at the attribute level of boolean correlated patterns [Kim

et al. 2004]. In this section, we analyze redundant knowledge at both the attribute level and the interval level of QCPs and then propose effective techniques to remove the redundancy.

7.1 Redundancy Elimination at the Attribute Level

Let $\mathcal{P}$ be a set of QCPs. Redundancy exists within $\mathcal{P}$, if some QCP in $\mathcal{P}$ has similar or the same all-confidence value as that of its super-pattern. For example, given a pattern $X = \texttt{education}[2,2]\texttt{salary}[2000,2500]$ and its super-pattern $Y = \texttt{education}[2,2]\texttt{marital\_status}[1,1]\texttt{salary}[2000,2500]$, if $allconf(X) = allconf(Y)$, then X is redundant because Y already conveys the information that X carries. Since X and Y have different attribute sets, we define the pattern X as a *redundant QCP at the attribute level* as follows.

Definition 7. (Redundant QCP at the Attribute Level) *$X \in \mathcal{P}$ is* redundant *if there exists some $Y \in \mathcal{P}$ such that, $X \subset Y$ and $allconf(X) = allconf(Y)$.*

According to Definition 7, we further define the concept of *all-confidence-closed QCP* to formalize non-redundant QCPs.

Definition 8. (All-Confidence-Closed QCP) *A pattern $X \in \mathcal{P}$ is called an* all-confidence-closed QCP *if there does not exist any $Y \in \mathcal{P}$ such that $Y \supset X$ and $allconf(Y) = allconf(X)$.*

In subsequent discussions, we simply say that a pattern X is *all-confidence-closed* if X is an all-confidence-closed QCP.

Now, we investigate some properties of all-confidence-closed QCPs and develop an efficient algorithm that computes the set of all-confidence-closed QCPs.

We first define an ordering, denoted as $\prec$, among the attribute sets. Let u and v be two nodes in T_{attr} and X and Y be the two attribute sets represented by u and v, respectively. We define $\prec$ as the prefix order on the attribute prefix tree T_{attr} as follows. $X \prec Y$ (or equivalently $Y \succ X$) if and only if u is visited before v in a pre-order traversal of T_{attr}.

Property 13. *Given a QCP X, if there exists an all-confidence-closed QCP Y, such that $Y \prec X$, $Y \supset X$ and $allconf(Y) = allconf(X)$, then X is not all-confidence-closed, and $\forall Z$, where $Z \supset X$ and $Z \succ X$, Z is not all-confidence-closed.*

Proof. By Definition 8, X is not all-confidence-closed due to the existence of Y. Since $Y \supset X$, we have $supp(Y) \leq supp(X)$ and $maxsupp(Y) \geq maxsupp(X)$. By Definition 4, we have $allconf(Y) \leq allconf(X)$. Since $allconf(Y) = allconf(X)$, we have $supp(Y) = supp(X)$ and $maxsupp(Y) = maxsupp(X)$.

Assume to the contrary that Z is all-confidence-closed. Let $x[l_x, u_x]$ be the item in X such that the attribute x is lexicographically ordered before all other attributes in X. Since $Y \prec X$ and $Y \supset X$, there exists an item $y[l_y, u_y] \in (Y \backslash X)$ such that y is lexicographically ordered before x. Thus, $\forall Z \supset X$ and $Z \succ X$, y is not an attribute in Z. Moreover, since $supp(Y) = supp(X)$, $y[l_y, u_y]$ must be supported by every transaction that supports X, which implies $supp(Z) = supp(Z \cup \{y[l_y, u_y]\})$. Since $Z \supset X$, we have $maxsupp(X) \leq maxsupp(Z)$. Since $y[l_y, u_y] \in Y$ and $maxsupp(Y) = maxsupp(X)$, we have $supp(y[l_y, u_y]) \leq maxsupp(X) \leq maxsupp(Z)$. Therefore,

$maxsupp(Z \cup \{y[l_y, u_y]\}) = MAX\{maxsupp(Z), supp(y[l_y, u_y])\} = maxsupp(Z)$. It follows that $allconf(Z) = allconf(Z \cup \{y[l_y, u_y]\})$, which is a contradiction to the assumption that Z is all-confidence-closed. □

Property 13 is important, since it can be employed to perform effective pruning of the search space in mining all-confidence-closed QCPs. When we explore T_{attr} in the mining process, if we find any pattern X having a super-pattern that has been discovered as an all-confidence-closed QCP and has the same all-confidence value as that of X, we can immediately stop generating longer patterns from X, i.e., we ignore all descendants of X in T_{attr}.

We now describe a revised algorithm of *QCoMine* that makes use of Property 13 to mine all-confidence-closed QCPs.

RecurMine_Closed, as presented in Procedure 2, is a procedure that replaces *RecurMine* (Procedure 1) to mine the set of all-confidence-closed QCPs. As shown in Step 10, when producing a new QCP Z, we first invoke *ClosedCheck* to check whether Z has a super-pattern that has been discovered as an all-confidence-closed QCP and has the same all-confidence value as that of Z. If *ClosedCheck*(Z) returns true, then Z is not all-confidence-closed and neither are all of its descendants in T_{attr} according to Property 13. In this case, Z is not to be produced. Otherwise, if Z has all-confidence no less than ς, then Z is either all-confidence-closed or is to be kept as an intermediate pattern to produce Z's super-patterns that are all-confidence-closed. In addition, if we find that X has the same all-confidence value as that of Z, we mark X as a redundant QCP according to Definition 7, such that X will not be outputted when the recursive call of *RecurMine_Closed* returns to X.

The efficient processing of *ClosedCheck* can be achieved using a hash table, which stores all discovered all-confidence-closed QCPs. There are two keys to the hash table. One is the support of an all-confidence-closed QCP. Another is the value of *maxsupp* of the QCP. If Z has a super-pattern Y in the hash table with the same all-confidence value, Y and Z must have the same value of *maxsupp*. To avoid collisions as much as possible, the hash function described in [Zaki and Hsiao 2002], which is based on the sum of the transaction IDs, can also be applied.

Given the set of all-confidence-closed QCPs, we are able to recover the whole set of QCPs based on the closure condition. Therefore, the set of all-confidence-closed QCPs is a lossless representation of the set of QCPs.

7.2 Redundancy Elimination at the Interval Level

In addition to the redundancy that exists among the QCPs with different attribute sets, redundancy also exists in QCPs that share the same attribute set. Due to the sub-interval enumeration as discussed in Section 6.2, it is possible that some QCPs of the same attribute set but different interval sets convey similar knowledge. As a simple example, given two patterns $X = \texttt{education}[2, 2]\texttt{salary}[2000, 2500]$ and $Y = \texttt{education}[2, 2]\texttt{salary}[2000, 2600]$, X is redundant with respect to Y, since it does not provide any new knowledge in the presence of Y. The underlying reason that we define X as a redundant pattern rather than Y is as follows. Both X and Y are generated based on a combined interval produced by the supervised interval combining process. The combined intervals of each attribute represent a meaningful partitioning of the domain of the attribute with respect to another

Procedure 2 *RecurMine_Closed*(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 $X = x_1[i_{u_1}] \ldots x_{k-1}[i_{u_{k-1}}]x_k[i_{u_k}]$ and $Y = x_1[i_{u_1}] \ldots x_{k-1}[i_{u_{k-1}}]x_{k+1}[i_{v_k}]$;
8. **if** $(\frac{supp(X)}{maxsupp(Y)} \geq \varsigma$ and $\frac{supp(Y)}{maxsupp(X)} \geq \varsigma)$
9. Let $Z = x_1[i_{u_1}] \ldots x_{k-1}[i_{u_{k-1}}]x_k[i_{u_k}]x_{k+1}[i_{v_k}]$;
10. **if** $((ClosedCheck(Z) = false)$ and $(allconf(Z) \geq \varsigma))$
11. Insert $\{i_{u_1}, \ldots, i_{u_{k-1}}, i_{u_k}, i_{v_k}\}$ into $T^w_{interval}$;
12. **if** $(allconf(X) = allconf(Z))$
13. Mark X as *redundant*;
14. Delete $T^u_{interval}$;
15. **for each** child node w of u **do**
16. *RecurMine_Closed*$(w, T_{attr}, G, k+1)$;

attribute. Therefore, since the sub-interval of the attribute `salary` in Y is larger than that in X (i.e., the sub-interval of `salary` in Y is closer to the combined interval from which the sub-interval is derived), Y is more desirable than X. We now formalize the definition of redundancy at the interval level as follows.

Definition 9. (Redundant QCP at the Interval Level) $X \in \mathcal{P}$ *is* redundant *if there exists some* $Y \in \mathcal{P}$ *such that,* $Y \neq X$, $attr(Y) = attr(X)$ *and* $\forall i_x \in interval(X)$, $i_x \sqsubseteq i'_x$, *where* $i'_x \in interval(Y)$ *is the interval of the attribute* x *in* Y.

The redundancy removal at the interval level can be efficiently performed by checking the intervals of the patterns when we output the QCPs. After removing the redundant QCPs at the interval level, we obtain a more concise set of QCPs having more meaningful intervals.

8. PERFORMANCE EVALUATION

We first evaluate the performance of our approach of mining correlations from quantitative databases on both real and synthetic datasets. Then we study an application of QCPs to the problem of classification on a number of real datasets. We conduct all experiments on an AMD Opteron 248 with 8GB RAM, running Linux 64-bit.

8.1 Performance on Real Datasets

We use three real datasets from the commonly used UCI machine learning repository [Asuncion and Newman 2007]. Table IV 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 parentheses. The detailed information of these datasets can be found in [Asuncion and Newman 2007]. These datasets possess some representative characteristics as follows: *image* consists of a moderate number of attributes and transactions; *spambase* has a large number of quantitative attributes; and *covtype* is the largest real dataset in the UCI repository and it has a large number of attributes.

Table IV. Dataset Description

Dataset	Transactions	Attributes (Quantitative)	Maximum Base Intervals
image	2,310	20(19)	377
spambase	4,601	58(57)	761
covtype	581,012	55(10)	700

8.1.1 Performance of QCoMine

We first evaluate the performance of our algorithm QCoMine, which mines the set of all QCPs for a given dataset. The efficiency of QCoMine and the quality of our QCPs are based on three major components that constitute the algorithm 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 that studies mining correlations from quantitative databases, we study the impact of these three components on the performance of our approach.

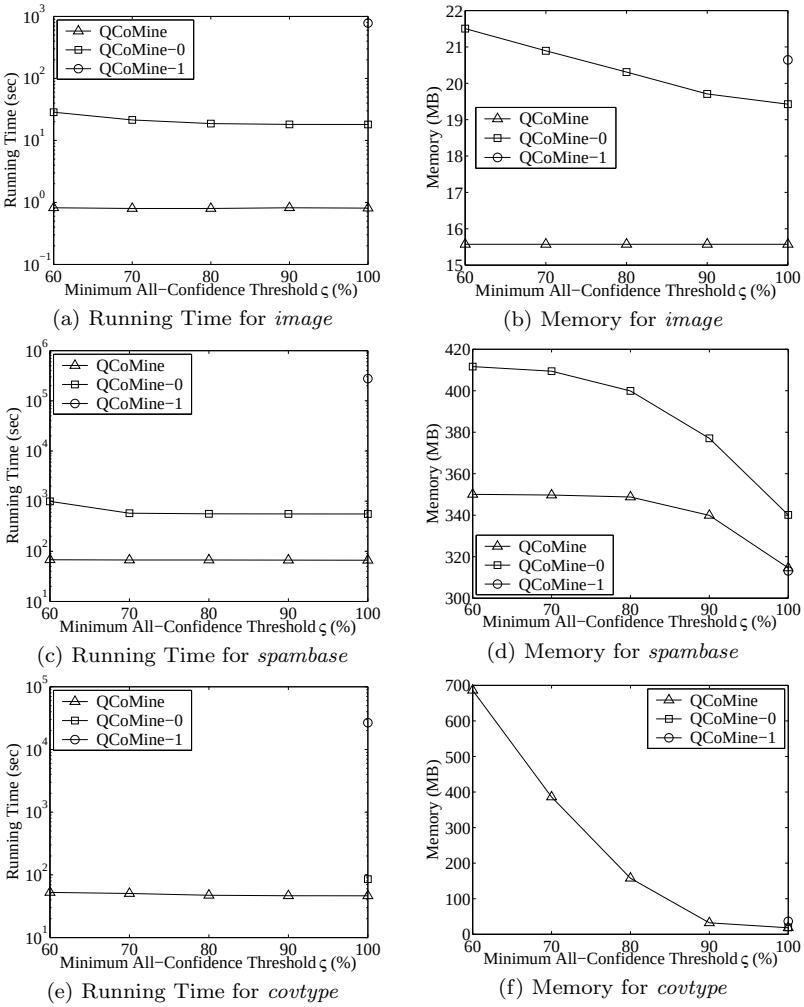
Our study is based on the three variants of our algorithm as follows:

(a) *QCoMine*, which applies the interval combining method and sets μ according to Equation (2);
(b) *QCoMine-0*, which applies the interval combining method and sets μ as 0;
(c) *QCoMine-1*, which does not apply the interval combining method and sets μ according to Equation (2). Since the interval combining method is not applied, the sub-interval enumeration is based on the whole domain of an attribute instead of the set of combined intervals.

We vary the value of all-confidence from $\varsigma = 60\%$ to $\varsigma = 100\%$.

8.1.1.1 Effect of Supervised Interval Combining

When the interval combining method is not applied, we are only able to obtain the results at $\varsigma = 100\%$, as shown by QCoMine-1 in Figures 3(a) to 3(f), while QCoMine-1 runs out of memory in all other cases. QCoMine-1 is inefficient because when we allow the interval of an attribute to become too trivial, the patterns easily gain all-confidence greater than ς by co-occurrence in the database. The number of patterns obtained by QCoMine-1 is many orders of magnitude larger than that



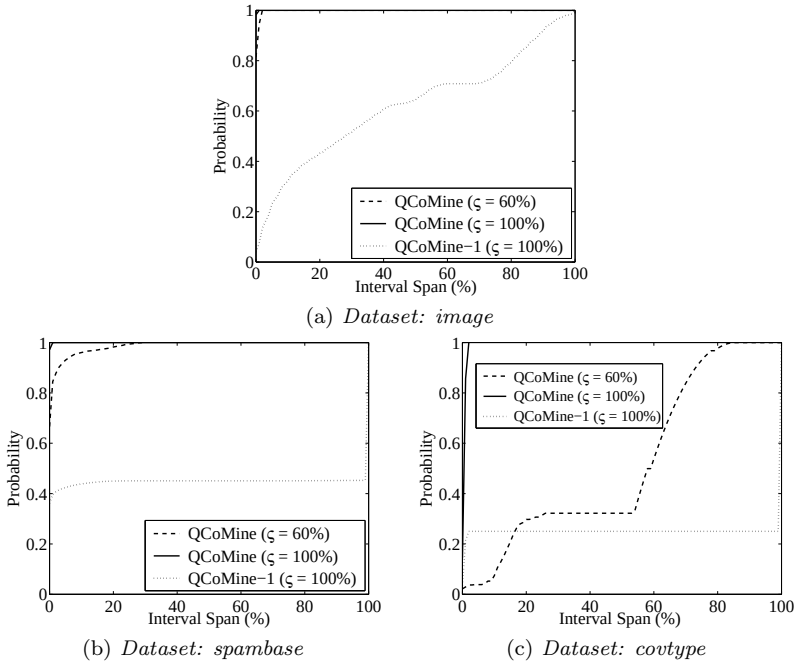
(a) Running Time for *image*
(b) Memory for *image*
(c) Running Time for *spambase*
(d) Memory for *spambase*
(e) Running Time for *covtype*
(f) Memory for *covtype*

Fig. 3. Running Time and Memory Consumption of QCoMine on Real Datasets

obtained by QCoMine and the difference increases rapidly for smaller ς (the memory is used up by QCoMine-1 for smaller ς as a result).

We then examine whether our interval combining method can effectively avoid generating trivial intervals. For this purpose, we define the *span* of an interval, $[l, u]$, of an attribute as the fraction, $\frac{u-l}{n}$, where n is the number of base intervals of the attribute. For example, if `age` has 100 base intervals, the span of the interval $[20, 80]$ is 60%.

Figures 4(a) to 4(c) report the cumulative probability distribution of the maximum span of the intervals in the patterns obtained by QCoMine-1 and QCoMine at $\varsigma = 100\%$, where we also show the results for QCoMine at $\varsigma = 60\%$ for reference. The figures show that all patterns returned by QCoMine consist of intervals with small spans. For all datasets, all the patterns returned by QCoMine consist of



(a) *Dataset: image*

(b) *Dataset: spambase*

(c) *Dataset: covtype*

Fig. 4. Cumulative Probability Distribution of Interval Span in QCPs on Real Datasets

intervals with a maximal span of less than 2% at $\varsigma = 100\%$. On the contrary, for the patterns returned by QCoMine-1, about 40% for *image*, 60% for *spambase* and 80% for *covtype* consist of intervals with very large spans. Particularly for *spambase* and *covtype*, the large span is tantamount to the entire domain of the attributes and simply trivial. Obviously, those patterns are returned as high all-confidence patterns simply due to the co-occurrence of the items in the datasets, since items with a large interval span also have a large support value.

From the above-mentioned results, we confirm that our supervised interval combining method is effective in defining more meaningful intervals and thus it avoids an overwhelming number of trivial patterns being mined, which is essential for efficient usage of memory and CPU resources.

8.1.1.2 Effect of Normalized Mutual Information

The benefit of utilizing NMI as a pruning tool is clearly revealed by the performance difference between QCoMine and QCoMine-0 shown in Figure 3. QCoMine is over an order of magnitude faster and consumes significantly less memory than QCoMine-0 for both *image* and *spambase*, while QCoMine-0 runs out of memory for $\varsigma \leq 90\%$ for *covtype*.

The number of patterns returned by QCoMine-0 is many orders of magnitude (up to four orders for *image* and three orders for *spambase*) more than that by QCoMine. This also explains why QCoMine-0 runs out of memory for *covtype*.

We examine the patterns obtained and find that the extra patterns returned by QCoMine-0 are shown to consist of attributes with large interval spans. Note that

QCoMine-0 also adopts our interval combining method; however, we emphasize that 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 by QCoMine. For example, we find that 60% of the extra patterns returned by QCoMine-0 for *image* consist of intervals with a span over 90% (i.e., almost the entire domain), while 80% of the extra patterns consist of intervals with a span over 50%.

To sum up, the results demonstrate the effectiveness of NMI both as a measure for correlation and as a tool for pruning unpromising search spaces.

8.1.1.3 Effect of All-Confidence

The effect of all-confidence is shown by the number of QCPs obtained by QCoMine. As shown in Figures 5(a) to 5(c), the number of QCPs decreases considerably when ς increases from 60% to 100%. (Figure 5 also shows the results of QCoMine-c and QCoMine-r, which are discussed in the following subsection when we evaluate the performance of redundancy elimination.)

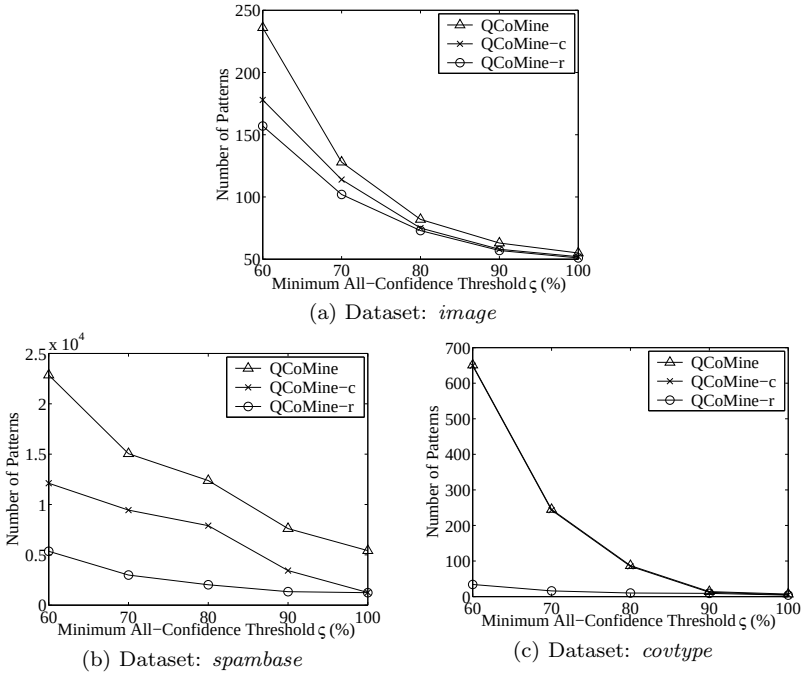
However, the running time of QCoMine does not increase with the increase in the number of QCPs. Figure 3 shows that the running time of QCoMine is very stable for different values of ς. This is because over 95% of the time is spent on the following three processes: the reading of the dataset from the disk, the interval combining, and the generation of the 2-patterns. These three processes are inevitable in the overall mining process. Therefore, the stabilization in the running time for different values of ς in fact reflects the pruning power of all-confidence, since QCoMine only spends a small amount of the time generating the QCPs when the pruning starts.

8.1.2 Performance of Redundancy Elimination

We now evaluate the effectiveness of our approach in eliminating the redundancy within the set of QCPs obtained by QCoMine. In this experiment, we denote *QCoMine-c* as the algorithm for mining the all-confidence-closed QCPs, which conducts redundancy elimination at the attribute level. Then, we extend QCoMine-c to remove the redundancy at the interval level and denote the corresponding algorithm as *QCoMine-r*.

Figures 5(a) to 5(c) report the number of patterns obtained at different values of ς by QCoMine, QCoMine-c and QCoMine-r, on the three datasets. The results show that a significant number of patterns are redundant for *spambase* and *covtype*. The number of non-redundant QCPs (represented by QCoMine-r) is about 5-6 times and up to 20 times less than the whole set of QCPs (represented by QCoMine) for *spambase* and *covtype*, respectively. The QCPs from *image* have less redundancy and we record a reduction of 1.5 times in the number of patterns.

It is interesting to see that the three datasets contain very different degrees of redundancy at the attribute level and at the interval level. Figure 5(a) shows that more redundant patterns from *image* are at the attribute level, while the majority



(a) Dataset: *image*

(b) Dataset: *spambase*

(c) Dataset: *covtype*

Fig. 5. Number of Patterns on Real Datasets

of the redundant patterns from *covtype* exist at the interval level as shown in Figure 5(c). This behavior of the datasets can be explained by the span distributions of the intervals of their patterns, which are reported in Figures 4(a) to 4(c). For the *image* dataset, as shown in Figure 4(a), the span of the intervals for all patterns is small for all values of ς; thus, the number of patterns due to redundant intervals is also small. On the contrary, for the *covtype* dataset, as shown in Figure 4(c), there are some patterns with large interval span when ς is small; consequently, more patterns are redundant at the interval level.

We also find that the number of redundant patterns at the attribute level for *covtype* is very small, i.e., almost all QCPs are all-confidence-closed. This can be explained by the large size of the *covtype* dataset and the definition of all-confidence-closed QCPs. Definition 8 implies that if a QCP X is not all-confidence-closed, then there must exist a super-pattern of X such that *every* transaction supporting X must also support the super-pattern of X. It is easy to see that this condition becomes difficult to satisfy when the number of transactions in the dataset is large.

The *spambase* dataset is not a large dataset and the span of the intervals of its patterns is larger than that from *image* but smaller than that from *covtype*. Therefore, we can see from Figure 5 that there is a more even distribution of redundant patterns at the attribute and interval levels. Note that the *spambase* dataset has a much larger number of QCPs than the other datasets. This is mainly because *spambase* has a larger number of quantitative attributes, which results in a more serious combinatorial explosion at the interval level.

Overall, the results of this experiment demonstrate the effectiveness of our *bi-*

level redundancy removal approach; that is, the redundancy is guaranteed to be removed at one level or another or both.

The running time and memory consumption of both QCoMine-c and QCoMine-r are almost the same as those of QCoMine shown in Figure 3. The difference in the running time is small, since the processes of reading the data from the disk and combining the intervals dominate. The memory consumption is also about the same, because an extra hashtable is used but fewer patterns are processed, which compensate for each other. Thus, we do not present the figures of the detailed results.

8.1.3 Quantitative Correlated Patterns vs. Quantitative Frequent Patterns

To further justify the feasibility of our approach for mining quantitative databases, we demonstrate the high complexity of mining *Quantitative Frequent Patterns* (*QFPs*) in this section.

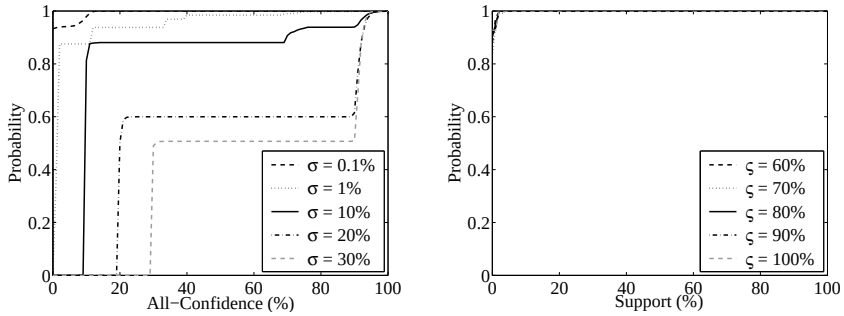
We implement the algorithm proposed by [Srikant and Agrawal 1996] using the same prefix tree structure and the *diffset* [Zaki and Gouda 2003] as used in QCoMine and denote this algorithm as MFP in this experiment. We test five settings of *minimum support threshold*, $\sigma = 0.1\%, 1\%, 10\%, 20\%, 30\%$, for MFP. Since MFP uses a *maximum support threshold*, σ_m, to control the span of a combined interval, we set $\sigma_m = 1.1\sigma$, which means that the support of a combined interval is at most 1.1 times of the minimum support threshold.

Figure 6(a) presents the cumulative probability distribution of all-confidence over the patterns obtained by MFP from *image*. When σ is small ($\leq 10\%$), around 90% of the patterns have very low all-confidence values of less than 10% (i.e., they are uncorrelated patterns). When $\sigma = 20\%$, there are still 60% of the patterns having all-confidence of only 20%. Although half of the patterns have all-confidence greater than 90% when $\sigma = 30\%$, 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 6(b), shows that most of the QCPs do not have high support (i.e., they are not commonsense patterns). In fact, many of the QCPs are rare patterns that are hidden and not easy to be discovered, but they are significant patterns 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.

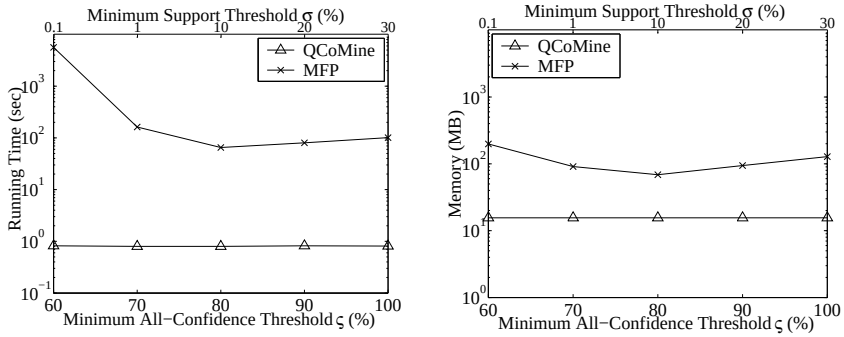
We also show the running time and memory consumption of MFP at each σ (as indicated by the upper x-axis) and QCoMine at each ς (as indicated by the lower x-axis) in Figures 6(c) and 6(d). Although QCoMine and MFP are incomparable in terms of running time and memory consumption due to different parameters of σ and ς, the figures do indicate that mining QCPs is much more stable in the use of resources than mining QFPs.

We do not present the results of MFP for *spambase* and *covtype*, because MFP runs out of memory for all values of σ, even when σ_m is set to be almost the same as σ. At the point that the memory is exhausted, MFP already returns millions of patterns that occupy over 20GB of disk space (for each σ). The massive number of generated patterns not only results in high memory consumption, but also gives



(a) Cumulative Probability Distribution of All-Confidence of QFPs

(b) Cumulative Probability Distribution of Support of QCPs



(c) Running Time

(d) Memory

Fig. 6. Quantitative Correlated Patterns vs. Quantitative Frequent Patterns for *image*

rise to difficulties in further analysis of the patterns. On the contrary, QCoMine obtains impressive results for both *spambase* and *covtype* as shown in Figures 3(c) to 3(f), which confirms the effectiveness and the efficiency of mining QCPs over mining QFPs.

We emphasize 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 bi-level pruning technique, as evidenced by the poor performance of QCoMine-0 and QCoMine-1 in Section 8.1.

8.2 Performance on Synthetic Datasets

We use the synthetic datasets that are generated by the IBM Quest Synthetic Data Generator for Classification [IBM Quest]. There are six quantitative attributes: `salary`, `commission`, `age`, `hvalue`, `hyears`, and `loan`. In addition, there are six categorical attributes: `elevel`, `car`, `zipcode`, `F1`, `F2`, and `F3`, where `F1`, `F2`, and `F3` correspond to Functions 1-3 described in [Agrawal et al. 1993b]. We generate five datasets of sizes from 200K to 1,000K transactions for carrying out a scalability test on QCoMine.

8.2.1 Effectiveness of the Supervised Interval Combining Method

We first justify the meaningfulness of the intervals produced by the supervised interval combining method. Here, we only present the combined intervals on the dataset with 800K transactions, since the results for other datasets are similar. There are several generative rules between the attributes in the synthetic generator as follows.

(1) $(\texttt{salary} \geq 75K) \Rightarrow (\texttt{commission} = 0)$; otherwise, the values of `commission` are uniformly distributed from $10K$ to $75K$.

Our result shows that none of the base intervals of `commission` are combined with respect to `salary`. However, this result perfectly conforms to the generative rule of `commission` shown above. The special `commission` value "0" should form a single interval $[0, 0]$ because "0" is generated according to a specific range of `salary`. Other base intervals of `commission` are not combined since other values of `commission` are uniformly distributed regardless of the values of `salary` and hence they are uncorrelated.

On the other hand, one of the combined intervals of `salary` with respect to `commission` is $[75K, 150K]$ and all other base intervals of `salary` are not combined, where $150K$ is the maximum value of `salary`. The combined intervals are meaningful, since the interval $[75K, 150K]$ exactly reflects the above generative rule, while other base intervals of `salary` do not have any effect on the values of `commission`.

(2) The values of `hvalue` are uniformly distributed from $50000k$ to $150000k$, where $k \in \{1, \ldots, 10\}$ are the values of `zipcode`.

The set of cutting points of the combined intervals of `hvalue` with respect to `zipcode` are $\{50K, 100K, 150K, \ldots, 1500K\}$, which is coincident with the generative rule of `hvalue`. Essentially, the values of `hvalue` are uniformly distributed within a set of overlapping intervals, i.e., $\{[50K, 150K], [100K, 300K], \ldots, [500K, 1500K]\}$, which further depends on the values of `zipcode`. Thus, the combined intervals of `hvalue` produced by our supervised interval combining method match the generative rule precisely.

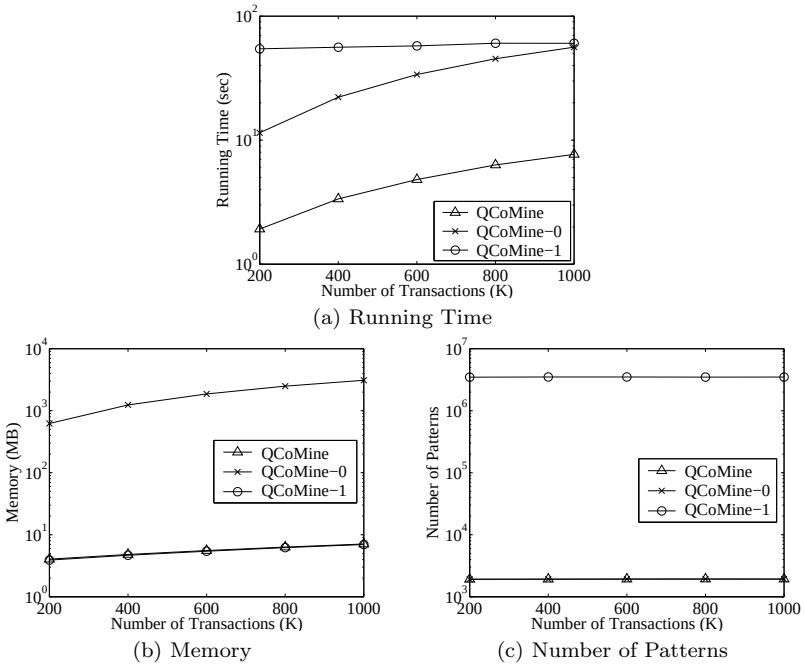
(3) $(\texttt{age} < 40) \vee (\texttt{age} \geq 60) \Rightarrow (\texttt{F1} = 1)$, otherwise, $\texttt{F1} = 0$.

The set of combined intervals of `age` with respect to `F1` is $\{[20, 39], [40, 59], [60, 80]\}$. This result once again shows that our supervised interval combining method perfectly captures the underlying dependency of the attributes.

8.2.2 Scalability of QCoMine

We vary the number of transactions in the synthetic datasets to test the scalability of QCoMine. We fix $\varsigma = 60\%$ for all datasets. The results are shown in Figure 7. As shown in Figures 7(a) and 7(b), QCoMine is very efficient for all dataset sizes and the performance degrades only slightly when the number of transactions increases from 200K to 1,000K. For all dataset sizes tested, QCoMine takes less than eight seconds to complete the entire mining process, while the memory consumption remains very stable at around 7MB.

The results of this experiment also reveal a very important finding on the use of NMI as a pruning tool and a measure of dependency between the attributes.



(a) Running Time

(b) Memory

(c) Number of Patterns

Fig. 7. Running Time, Memory Consumption and Number of Patterns of QCoMine on Synthetic Datasets

Figure 7(c) shows that using NMI (cf. the QCoMine line) obtains almost the same set of QCPs as without using NMI (cf. the QCoMine-0 line). This implies that the patterns pruned by NMI in QCoMine are indeed uncorrelated patterns, since they also have low all-confidence; thus, the effectiveness of NMI as a dependency measure is clearly demonstrated. However, Figures 7(a) and 7(b) show that both the running time and memory consumption of QCoMine-0 are over an order of magnitude greater than those of QCoMine; thus, the pruning power of NMI is also clearly demonstrated. The results also show the great effect of the interval combining on controlling the running time and the number of patterns produced, as shown in Figures 7(a) and 7(c).

8.2.3 Examples of QCPs Obtained

After getting rid of the redundancy in the QCPs, we obtain the following two QCPs for all the synthetic datasets:

—`salary`[75K, 150K]`commission`[0, 0]: $allconf = 1$
—`age`[40, 59]`F1`[0, 0]: $allconf = 1$

These two QCPs exactly demonstrate the first and the third generative rules in the generator in Section 8.2.1. Therefore, the result verifies that the concept of QCP is able to capture correctly the dependency between the attributes in the database. Note that the QCPs fail to capture the second generative rule because there are many combined intervals for the two attributes `hvalue` and `zipcode` resulting from their generative rule. This results in relatively low all-confidence for

most of the patterns containing these two attributes. We remark that the above two patterns cannot be obtained by mining QFPs. This is because the interval combining method in MFP is not able to produce these meaningful intervals and the support measure is not effective in capturing the underlying dependency of the attributes.

8.3 Application of QCPs for Classification

According to the semantics of the all-confidence measure, we know that QCPs essentially represent a set of association rules that have confidence of no less than the predefined threshold. Different from the conventional association rules, these association rules have attributes that are pairwise-correlated as defined by NMI. Therefore, QCPs can be applied to wherever association rules can be applied. In this paper, we study an application of QCPs for the problem of *classification*, which is a well-recognized application of association rules, as well as an objective method for evaluating the quality of association rules [Mutter et al. 2004].

An association rule that can be used to predict the class label of the unlabeled data has the class attribute on the right-hand side. Such an association rule is called a *class association rule*. The classifier built on the class association rules is called an *associative classifier* [Hu et al. 1999; Li et al. 2001; Yin and Han 2003]. Associative classifiers first mine a set of class association rules from the training data and then use them to predict the class label of the unlabeled data. It has been shown that associative classifiers achieve higher classification accuracy than the decision-tree-based approach, C4.5 [Quinlan 1993].

In our experiment, we generate a set of class association rules from the QCPs for classification. Ten datasets from the UCI repository are tested. We compare the accuracy of classification using QCPs with that of three state-of-the-art associative classifiers: *CBA* [Hu et al. 1999], *CMAR* [Li et al. 2001], and *CPAR* [Yin and Han 2003]. In our implementation, instead of building a new classifier, we simply feed the class association rules generated from QCPs into the CPAR classifier. That is, we replace the rules generated by CPAR with the rules generated from QCPs. In this way, we are able to see clearly the effect of QCPs as prediction rules on the classification accuracy and demonstrate whether the improvement only comes from the use of QCPs. We denote the classifier built on QCPs as *QCP-Classifier* in the experiment.

We use the default settings for each associative classifier as specified in their papers. For QCoMine, we use $\varsigma = 0.2$. The relatively low setting of ς is because the measure of all-confidence treats all attributes equally in a QCP. However, for the purpose of classification, the class attribute must appear on the right-hand side of the association rule. Therefore, if we set a high ς in QCoMine, we will not be able to get enough association rules for the classifier. We set μ to be the smallest NMI value between the class attribute and all the other attributes so that each attribute has a chance to be included in a QCP and later in a class association rule for prediction.

Table V shows the classification accuracy of the four classifiers. The accuracy on each dataset is the average of the 10-fold cross-validations. Among the ten datasets, QCP-Classifier achieves the highest accuracy for seven datasets. Moreover, its average accuracy is the highest among the four classifiers. The accuracy

obtained by QCP-Classifier is significantly higher than that of CBA and CPAR and is approximately 1% higher than that of CMAR, which is the highest among all the associative classifiers. The higher accuracy of QCP-Classifier is because the attributes in a QCP are mutually correlated as measured by NMI. The measure of all-confidence further enhances the correlation between the class label and the values of the other attributes. Consequently, the association rules generated from QCPs utilize those attributes that are closely correlated to the class attribute to perform the prediction. On the other hand, the association rules generated by the associative classifiers suffer from the problem of over-fitting the training data. As a result, the attributes in the rules may not truly decide the value of the class attribute. Therefore, the results verify the high quality of QCPs in capturing the dependency between the attributes.

Table V. Classification Accuracy on Ten Real Datasets

Dataset	CBA	CMAR	CPAR	QCP-Classifier
australian	84.9	86.1	**86.2**	**86.2**
cleve	82.8	82.2	80.5	**84.2**
crx	84.7	84.9	85.9	**86.1**
heart	81.9	82.2	**82.6**	81.9
hepatitis	**81.8**	80.5	78.7	81.3
horse	82.1	82.6	82.1	**84**
ionosphere	92.3	91.5	**92.6**	90.6
iris	94.7	94	94.7	**95.3**
labor	86.3	89.7	88	**90**
lymph	77.8	83.1	82.3	**85.9**
Average	84.93	85.68	85.36	**86.55**

We also present the average running time and the average number of class association rules used in CPAR and QCP-Classifier in Table VI. We only compare QCP-Classifier with CPAR since it is shown to be the most efficient and use the least number of rules among the three associative classifiers [Yin and Han 2003]. We find that, although the number of rules used in QCP-Classifier is slightly larger than that in CPAR, the running time of QCP-Classifier is still over two times faster than that of CPAR. This shows another benefit of using QCPs: the efficient generation of QCPs also facilitates its application.

Table VI. Running Time and Number of Rules: CPAR and QCP-Classifier

	CPAR	QCP-Classifier
Running Time (msec.)	22.9	11
Number of Rules	119	128

We further investigate the two sets of class association rules used in CPAR and QCP-Classifier. We find that the two sets of rules are quite different. On average, only less than 10% of the rules are in common. The length of the rules in CPAR are two times longer than that in QCP-Classifier. This is because the attributes in a QCP are required to be mutually dependent. As a result, the length of the association rules generated from QCPs is small, usually less than five. On the other

hand, the conventional association rules are measured by support and confidence. Therefore, they tend to have longer length. However, these long rules can easily over-fit the training data.

Table VII. Example Rules in CPAR and QCP-Classifier

Dataset	Rule
australian	(1) A5[ff] A10[34, 67] ⇒ Class[-] (2) A3[15, 28] A4[g] A9[t] A14[50000, 100001] ⇒ Class[+]
cleve	(1) Age[29, 53] Max_Heart_Rate[137, 202] Thal[normal] ⇒ Class[healthy] (2) Sex[male] Resting_Blood_Pressure[94, 147] Resting_Ecg[normal] Number_of_Vessels_Colored[0, 1] ⇒ Class[sick]
crx	(1) A9[t] A11[34, 67] ⇒ Class[-] (2) A7[z] A8[14.5, 28.5] A10[f] A13[p] ⇒ Class[+]
heart	(1) Resting_Blood_Pressure[94, 147] Serum_Cholestoral[345, 564] ⇒ Class[healthy] (2) Age[29, 53] Serum_Cholestoral[126, 345] Max_Heart_Rate[71, 137] Exercise_Induced_Angina[true] ⇒ Class[sick]
hepatitis	(1) Spiders[yes] Varices[yes] ⇒ Class[live] (2) Anorexia[no] Liver_Big[no] Protime[50, 100] Histology[no] ⇒ Class[die]
horse	(1) Abdomen[distended_small_intestine] ⇒ Class[surgical_lesion] (2) Temperature_of_Extremities[cold] Capillary_Refill_Time[more_than_3_seconds] Nasogastric_Tube[none] Outcome[lived] ⇒ Class[no_surgical_lesion]
ionosphere	(1) Antenna6[-0.33, 0.33] ⇒ Class[good] (2) Antenna7[-0.6, -0.2] Antenna28[-0.33, 0.33] ⇒ Class[bad]
iris	(1) Petal_Width[0.9, 1.7] ⇒ Class[iris_versicolour] (2) Petal_Length[1, 2.97] ⇒ Class[iris_virginica]
labor	(1) Cost_of_Living_Adjustment[tc] Contribution_to_Dental_Plan[full] ⇒ Class[good] (2) Wage_Increase_in_First_Year[4.5, 7] Standby_Pay[2, 8] ⇒ Class[bad]
lymph	(1) Defect_in_Node[lacunar_marginal] Dislocation[no] ⇒ Class[metastases] (2) Block_of_Affere[no] By_Pass[no] Early_Uptake_In[yes] Special_Forms[vesicles] ⇒ Class[malign_lymph]

Table VII lists some examples of class association rules used in CPAR and QCP-Classifier. Note that the names of the attributes in the *australian* and *crx* datasets are mapped to anonymous labels since the datasets concern credit card applications, which are confidential. For each dataset, the first rule is used in QCP-Classifier and the second is used in CPAR. They are used in the two respective classifiers to predict the class label of the same unlabeled instance. The first rule gives the correct label while the second gives the wrong one, although the confidence values of these two rules are close. We then investigate the values of NMI between the pairwise attributes in the rules. Take the two rules from the *lymph* dataset for example. We find that the minimum NMI of the first rule is 0.05 while that of the second rule is 0.0004. This result reveals the major advantage of QCPs over conventional association rules: QCPs are able to capture the dependency of attributes while conventional association rules fail to. As verified by our experimental results, higher classification accuracy is achieved when the attributes that are highly correlated to the class attribute are used to perform the prediction.

9. RELATED WORK

The existing research on mining quantitative databases has mainly focused on mining knowledge in the form of association rules. This was first studied by [Piatetsky-Shapiro 1991] with both sides of the rule restricted to a single attribute with its

interval. [Srikant and Agrawal 1996] generalized the work by allowing multiple attributes on both sides of the rule.

We are also aware of mining two variants of association rules in quantitative databases. The first is the optimized association rules [Fukuda et al. 2001; Brin et al. 1999; Rastogi and Shim 2002], which contain certain uninstantiated attributes and the mining problem is to determine values for the uninstantiated attributes such that one measure (e.g., support, confidence or gain) is maximized and another measure satisfies a threshold. This type of work focuses on finding the optimal values of certain given attributes instead of mining correlations among all the attributes in the database. The second is the association rules that are based on statistics [Aumann and Lindell 2003; Webb 2001; Zhang et al. 2004], in which the right side of the rule is a statistical measure (e.g., mean, variance) or some aggregate function (e.g., min, max) of a quantitative attribute, rather than the interval information of the attributes.

[Wang et al. 1998] proposed 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 are employed to further guide the generation of QCPs.

In mining correlations from boolean databases, [Brin et al. 1997] generalized the association rules to correlations and introduced the correlation measures, χ^2 and *interest*. [Motwani et al. 2001] proposed to mine highly correlated 2-patterns measured by a symmetric similarity between two boolean attributes. [Ma and Hellerstein 2001] proposed an m-pattern, of which any two subsets are mutually dependent measured by the conditional probability. [Omiecinski 2003] proposed two interesting measures, *all-confidence* and *bond*, both of which have the downward closure property. [Xiong et al. 2003; 2006] developed 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 [Lee et al. 2003] and [Kim et al. 2004], all-confidence is shown to be a better measure for correlations than χ^2 and *interest*, since all-confidence is not influenced by the co-absence of the items in the database, as are χ^2 and *interest*. Recently, [Xiong et al. 2006] proposed an efficient algorithm, called *TAPER*, for mining correlated pairs of items measured by the ϕ correlation coefficient. Later, [Zhang and Feigenbaum 2006] adopted the framework of TAPER and proposed a more efficient algorithm that uses min-hash functions [Cohen 1997; Motwani et al. 2001] to achieve greater pruning.

We are also aware of different proposals of normalized mutual information in the literature. For data clustering [Strehl 2002], the normalized mutual information of two attributes x and y is defined as $\frac{2 \cdot I(x;y)}{\max_{A \in \{1,\ldots,k\}}(H(A)) + \max_{B \in \{1,\ldots,g\}}(H(B))}$, where A represents possible cluster labels and B represents possible category labels. Normalized mutual information is also found in the area of image processing [Studholme et al. 1999], where it is defined as $\frac{H(x)+H(y)}{H(x,\ y)}$. In our recent work [Ke et al. 2006a], we defined normalized mutual information as $\frac{I(x;y)}{I(x;x)}$ for mining *quantitative association rules*. This definition is asymmetric, since an association rule is

an implication of the antecedent on the consequent.

Comparison with Our Prior Work. This paper substantially extends our prior work [Ke et al. 2006b] in several ways. First, this paper re-designs the supervised interval combining method, which now takes into account both input attributes simultaneously. This solves a major pitfall in the prior work [Ke et al. 2006b] that the order of input attributes greatly affects the output combined intervals. Second, this paper discusses the cross-support property of all-confidence and employs this property as another pruning strategy to improve the efficiency of mining QCPs further. Third, this paper studies non-redundant QCPs at both the attribute level and the interval level, which was not considered in the prior work. Effective and efficient techniques are presented for eliminating the redundancy. Finally, a more comprehensive empirical study is conducted to evaluate the performance of mining QCPs. In addition to enhancing the previous experimental evaluation, we conduct a new spectrum of experiments to justify the meaningfulness of the combined intervals, perform a scalability test on the QCoMine algorithm, as well as demonstrate the effectiveness and efficiency of redundancy elimination.

10. CONCLUSIONS

In this paper, we proposed the new notion of QCP, which achieves bi-level quality control of correlations based on NMI and all-confidence. We developed a supervised interval combining method to combine the intervals according to the dependency between the attributes. We then devised an efficient algorithm, QCoMine, to mine QCPs by utilizing NMI and all-confidence to perform bi-level pruning. We also proposed effective techniques to eliminate the redundancy existing within the set of QCPs at both attribute and interval levels.

Extensive experiments were conducted to examine various aspects of our approach. First, the results reveal that our interval combining method derives meaningful intervals and effectively eliminates the generation of trivial intervals. Second, the results demonstrate that NMI is both an effective measure of correlations and a powerful tool for pruning unpromising search spaces arising from uncorrelated patterns. In addition, the results show that all-confidence further ensures a stable performance for QCoMine as well as the quality of the QCPs obtained. Third, the results justify the effectiveness and efficiency of our techniques in removing the redundant QCPs. Fourth, the results show that QCoMine attains impressive speed but has small memory consumption, even when mining QFPs [Srikant and Agrawal 1996] becomes too expensive. More importantly, the QCPs obtained are shown to be more useful than the QFPs, since the QCPs are rare and correlated patterns while most of the QFPs are uncorrelated, common patterns. Finally, we applied QCPs to the problem of classification and showed that the classifier built from QCPs is able to achieve higher accuracy and is very efficient. Based on the solid empirical evidence, we conclude that mining QCPs, which is founded on the well-established information theory, is both an effective and efficient approach for mining quantitative databases.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their valuable comments, which have greatly helped improve the quality of the paper.

REFERENCES

Agrawal, R., Imielinski, T., and Swami, A. 1993a. Mining association rules between sets of items in large databases. In *SIGMOD '93: Proceedings of the 1993 ACM SIGMOD international conference on Management of data*. ACM Press, New York, NY, USA, 207–216.

Agrawal, R., Imielinski, T., and Swami, A. N. 1993b. Database mining: A performance perspective. *IEEE Trans. Knowl. Data Eng. 5,* 6, 914–925.

Asuncion, A. and Newman, D. 2007. UCI machine learning repository.

Aumann, Y. and Lindell, Y. 2003. A statistical theory for quantitative association rules. *J. Intell. Inf. Syst. 20,* 3, 255–283.

Brin, S., Motwani, R., and Silverstein, C. 1997. Beyond market baskets: generalizing association rules to correlations. In *SIGMOD '97: Proceedings of the 1997 ACM SIGMOD international conference on Management of data*. ACM Press, New York, NY, USA, 265–276.

Brin, S., Rastogi, R., and Shim, K. 1999. Mining optimized gain rules for numeric attributes. In *KDD '99: Proceedings of the 5th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM Press, New York, NY, USA, 135–144.

Chen, Z.-Y. and Liu, G.-H. 2005. Quantitative association rules mining methods with privacy-preserving. In *PDCAT '05: Proceedings of the 6th International Conference on Parallel and Distributed Computing Applications and Technologies*. IEEE Computer Society, Washington, DC, USA, 910–912.

Cohen, E. 1997. Size-estimation framework with applications to transitive closure and reachability. *J. Comput. Syst. Sci. 55,* 3, 441–453.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. 2001. *Introduction to algorithms*. The MIT Press.

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

Fukuda, T., Morimoto, Y., Morishita, S., and Tokuyama, T. 2001. Data mining with optimized two-dimensional association rules. *ACM Trans. Database Syst. 26,* 2, 179–213.

Hu, K., Lu, Y., Zhou, L., and Shi, C. 1999. Integrating classification and association rule mining: A concept lattice framework. In *RSFDGrC '99: Proceedings of the 7th International Workshop on New Directions in Rough Sets, Data Mining, and Granular-Soft Computing*. Springer-Verlag, London, UK, 443–447.

IBM Quest. IBM Quest Synthetic Data Generation Code for Classification. *http://www.almaden.ibm.com/cs/projects/iis/hdb/Projects/data_mining/mining.shtml*.

Ke, Y., Cheng, J., and Ng, W. 2006a. Mic framework: An information-theoretic approach to quantitative association rule mining. In *ICDE '06: Proceedings of the 22nd International Conference on Data Engineering (ICDE'06)*. IEEE Computer Society, Washington, DC, USA, 112.

Ke, Y., Cheng, J., and Ng, W. 2006b. Mining quantitative correlated patterns using an information-theoretic approach. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM Press, New York, NY, USA, 227–236.

Kim, W.-Y., Lee, Y.-K., and Han, J. 2004. CCMine: Efficient mining of confidence-closed correlated patterns. In *PAKDD '04: Proceedings of the 8th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*. 569–579.

Lee, Y.-K., Kim, W.-Y., Cai, Y. D., and Han, J. 2003. CoMine: Efficient mining of correlated patterns. In *ICDM '03: Proceedings of the 3rd IEEE International Conference on Data Mining*. IEEE Computer Society, Washington, DC, USA, 581.

Li, W., Han, J., and Pei, J. 2001. Cmar: Accurate and efficient classification based on multiple class-association rules. In *ICDM '01: Proceedings of the 2001 IEEE International Conference on Data Mining*. IEEE Computer Society, Washington, DC, USA, 369–376.

Ma, S. and Hellerstein, J. L. 2001. Mining mutually dependent patterns. In *ICDM '01: Proceedings of the 2001 IEEE International Conference on Data Mining*. IEEE Computer Society, Washington, DC, USA, 409–416.

Motwani, R., Cohen, E., Datar, M., Fujiwara, S., Gionis, A., Indyk, P., Ullman, J. D., and Yang, C. 2001. Finding interesting associations without support pruning. *IEEE Transactions on Knowledge and Data Engineering (special issue) 13,* 1, 64–78.

Mutter, S., Hall, M., and Frank, E. 2004. Using classification to evaluate the output of confidence-based association rule mining. In *Australian Conference on Artificial Intelligence*. 538–549.

Omiecinski, E. R. 2003. Alternative interest measures for mining associations in databases. *IEEE Transactions on Knowledge and Data Engineering 15,* 1, 57–69.

Pearson, K. 1904. On the theory of contingency and its relation to association and normal correlation. *Drapers' Company Research Memoirs, Biometric Series I*.

Piatetsky-Shapiro, G. 1991. Discovery, analysis, and presentation of strong rules. In *Knowledge Discovery in Databases*. AAAI/MIT Press, 229–248.

Quinlan, J. R. 1993. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers.

Rastogi, R. and Shim, K. 2002. Mining optimized association rules with categorical and numeric attributes. *IEEE Transactions on Knowledge and Data Engineering 14,* 1, 29–50.

Ruckert, U., Richter, L., and Kramer, S. 2004. Quantitative association rules based on half-spaces: An optimization approach. In *ICDM '04: Proceedings of the 4th IEEE International Conference on Data Mining (ICDM'04)*. IEEE Computer Society, Washington, DC, USA, 507–510.

Shannon, C. 1948. A mathematical theory of communication, i and ii. *The Bell System Technical Journal*, 379–423, 623–656.

Srikant, R. and Agrawal, R. 1996. Mining quantitative association rules in large relational tables. In *SIGMOD '96: Proceedings of the 1997 ACM SIGMOD international conference on Management of data*. 1–12.

Strehl, A. 2002. Relationship-based clustering and cluster ensembles for high-dimensional data mining. *PhD Thesis*.

Studholme, C., Hawkes, D., and Hill, D. 1999. An overlap invariant entropy measure of 3d medical image alignment. *Pattern Recognition*, 71–86.

Tan, P.-N., Kumar, V., and Srivastava, J. 2002. Selecting the right interestingness measure for association patterns. In *KDD '02: Proceedings of the 8th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM Press, New York, NY, USA, 32–41.

Wang, K., Tay, S. H. W., and Liu, B. 1998. Interestingness-based interval merger for numeric association rules. In *KDD '98: Proceedings of the 5th ACM SIGKDD international conference on Knowledge discovery and data mining*. 121–128.

Webb, G. I. 2001. Discovering associations with numeric variables. In *KDD '01: Proceedings of the 7th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM Press, New York, NY, USA, 383–388.

Xiong, H., Shekhar, S., Tan, P.-N., and Kumar, V. 2006. TAPER: A two-step approach for all-strong-pairs correlation query in large databases. *IEEE Transactions on Knowledge and Data Engineering 18,* 4, 493–508.

Xiong, H., Tan, P.-N., and Kumar, V. 2003. Mining strong affinity association patterns in data sets with skewed support distribution. In *ICDM '03: Proceedings of the 3rd IEEE International Conference on Data Mining*. IEEE Computer Society, Washington, DC, USA, 387.

Xiong, H., Tan, P.-N., and Kumar, V. 2006. Hyperclique pattern discovery. *Data Min. Knowl. Discov. 13,* 2, 219–242.

Yin, X. and Han, J. 2003. Cpar: Classification based on predictive association rules. In *SDM '03: Proceedings of the 3rd SIAM International Conference on Data Mining*. SIAM, San Francisco, CA, USA.

Zaki, M. J. and Gouda, K. 2003. Fast vertical mining using diffsets. In *KDD '03: Proceedings of the 9th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM Press, New York, NY, USA, 326–335.

Zaki, M. J. and Hsiao, C.-J. 2002. Charm: An efficient algorithm for closed itemset mining. In *SDM '02: Proceedings of the 2nd SIAM International Conference on Data Mining*.

Zhang, H., Padmanabhan, B., and Tuzhilin, A. 2004. On the discovery of significant statistical quantitative rules. In *KDD '04: Proceedings of the 10th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM Press, New York, NY, USA, 374–383.

Zhang, J. and Feigenbaum, J. 2006. Finding highly correlated pairs efficiently with powerful pruning. In *CIKM '06: Proceedings of the 15th ACM international conference on Information and knowledge management*. ACM Press, New York, NY, USA, 152–161.

...